
UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

# Um Protótipo para Desenvolvimento e Aplicação de Estratégias de Otimização em Sistemas xDSL Práticos

**Eduardo Lins de Medeiros**

**DM_04/2010**

UFPA/ITEC/PPGEE

Campus Universitário do Guamá

Belém-Pará-Brasil

2010

UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

# Um Protótipo para Desenvolvimento e Aplicação de Estratégias de Otimização em Sistemas xDSL Práticos

**Autor: Eduardo Lins de Medeiros**

**Orientador: Aldebaro Barreto da Rocha Klautau Júnior**

**Dissertação** submetida à Banca Examinadora do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Pará para obtenção do Grau de Mestre em Engenharia Elétrica. Área de concentração: **Telecomunicações**.

UFPA/ITEC/PPGEE

Campus Universitário do Guamá

Belém, PA

2010

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

# Um Protótipo para Desenvolvimento e Aplicação de Estratégias de Otimização em Sistemas xDSL Práticos

AUTOR: Eduardo Lins de Medeiros

DISSERTAÇÃO DE MESTRADO SUBMETIDA À AVALIAÇÃO DA BANCA EXAMINADORA APROVADA PELO COLEGIADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA, DA UNIVERSIDADE FEDERAL DO PARÁ E JULGADA ADEQUADA PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA ELÉTRICA NA ÁREA DE TELECOMUNICAÇÕES.

---

Prof. Dr. Aldebaro Barreto da Rocha Klautau Júnior
(Orientador - UFPA)

---

Prof. Dr. Antônio Jorge Gomes Abelém
(Membro - UFPA)

---

Prof. Dr. Joao Crisóstomo Weyl Albuquerque Costa
(Membro - UFPA)

---

Prof. Dr. Mauro Margalho Coutinho
(Membro - UNAMA)

UFPA/ITEC/PPGEE

*Este trabalho é dedicado à minha família e orientador.*

# Agradecimentos

Agradeço à minha família pelo apoio e amor incondicional que recebo desde que vim ao mundo.

Agradeço ao meu orientador, que me acompanha desde o início da graduação. Agradeço por sempre ter depositado confiança em mim, e principalmente pelo exemplo que dá, com a dedicação que persegue seus objetivos. Para minha carreira e para minha vida, foi muito enriquecedor ter sido seu orientado.

Agradeço aos meus amigos, principalmente aqueles do Laboratório de Processamento de Sinais e Laboratório de Eletromagnetismo Aplicado, que me acompanharam durante o mestrado.

Agradeço à Lilian pelo carinho e por ter me estimulado e apoiado durante a escrita desta dissertação.

Agradeço à equipe do projeto Ericsson UFA06, no Brasil e na Suécia, pelo apoio, suporte financeiro e cooperação técnica.

Finalmente, agradeço aos membros da banca pela cooperação na melhoria deste trabalho.

# Resumo

Este trabalho tem como objetivo o desenvolvimento de um protótipo para aplicação de estratégias de otimização em sistemas xDSL práticos. O protótipo descrito constitui-se de componentes de software e um *setup* experimental, capaz de gerar estímulos (tráfego e ruídos) para uma rede DSL completa, a fim de avaliar seu desempenho num ambiente controlado com equipamentos comerciais não modificados. Os resultados obtidos com o protótipo, bem como suas aplicações no campo de otimização de sistemas DSL são apresentados.

**Palavras chave:** Linha Digital de Assinante (DSL), Gerenciamento Dinâmico de Espectro (DSM), estratégias de otimização, estabilidade.

# Abstract

This work describes the development of a prototype for the application of optimization strategies for practical xDSL systems. The prototype described here is made of software components and an experimental setup, capable of generating IP traffic and noise signals, in order to evaluate the performance of a complete DSL network in a controlled environment, with commercial unmodified hardware. The results obtained with the prototype as well as its applications for optimization of practical xDSL systems are presented.

**Keywords:** Digital Subscriber Line (DSL), Dynamic Spectrum Management (DSM), optimization strategies, stability.

# Lista de Figuras

# Lista de Tabelas

# Lista de Siglas

**AAL5** - *ATM Adaptation Layer 5*

**ABR** - *Available Bit Rate*

**ADSL** - *Asymmetric Digital Subscriber Line*

**AMA** - *Automatic Margin Adaptation*

**ANSI** - *American National Standards Institute*

**API** - *Application programming Interface*

**ARP** - *Address Resolution Protocol*

**ATM** - *Asynchronous Transfer Mode*

**BER** - *Bit Error Rate*

**CBR** - *Constant Bit Rate*

**CO** - *Central Office*

**CPE** - *Costumer Premises Equipment*

**DLM** - *Dynamic Link Management*

**DMT** - *Discrete Multi-tone*

**DSL** - *Digital Subscriber Line*

**DSLAM** - *Digital Subscriber Line Access Multiplexer*

**DSM** - *Dynamic Spectrum Management*

**ETSI** - *European Telecommunications Standards Institute*

**FDD** - *Frequency-division Duplexing*

**FEC** - *Forward Error Correction*

**FEXT** - *Far-end Crosstalk*

**INP** - *Impulse Noise Protection*

**IP** - *Internet Protocol*

**ISB** - *Iterative Spectrum Balancing*

**ITU** - *International Telecommunication Union*

**IWF** - *Iterative Waterfilling*

**JPA** - *Java Persistence API*

**MAC** - *Media Access Control*

**NEXT** - *Near-end Crosstalk*

**OID** - *Object Identifier*

**PC** - *Personal Computer*

**PSD** - *Power Spectrum Density*

**RFI** - *Radio Frequency Interference*

**RT** - *Remote Terminal*

**SIMO** - *Single Input, Multiple Output*

**SNMP** - *Simple Network Management Protocol*

**SNR** - *Signal to Noise Ratio*

**SNRM** - *SNR Margin*

**SQL** - *Structured Query Language*

**SSM** - *Static Spectrum Management*

**UBR** - *Unspecified Bit Rate*

**VBR** - *Variable Bit Rate*

**VCI** - *Virtual Channel Identifier*

**VDSL** - *Very-high-bitrate Digital Subscriber Line*

**VLAN** - *Virtual Local Area Network*

**VPI** - *Virtual Path Identifier*

# Sumário

# Capítulo 1

# Introdução

## 1.1 Motivação

Nos últimos anos, com o crescimento da importância do acesso à internet em nossas vidas cotidianas, o desempenho das redes de acesso tem se tornado um fator muito relevante. Do ponto de vista dos clientes, maiores capacidades nos enlaces estão relacionados à uma maior rapidez no acesso ao grande número de informações disponíveis *online*. Economicamente, para os operadores dessas redes de acesso, um melhor desempenho nos enlaces garante a possibilidade de oferta de mais serviços, potencialmente mais consumidores e um maior valor agregado por enlace instalado.

Considerando os possíveis retornos em relação ao investimento, serviços baseados em linhas digitais de assinante (DSL) se destacam entre os competidores, já que a maior parte do investimento em infra-estrutura (os pares trançados de cobre) já se encontra há muito tempo disponível nas ruas, chegando de forma praticamente universal aos lares em áreas urbanas.

O desempenho de comunicações de banda larga sobre linhas digitais de assinante (DSL) é severamente limitado pela interferência de *crosstalk* (diafonia) proveniente de pares trançados adjacentes na rede de acesso [1, 2]. Por esta razão, técnicas de gerenciamento dinâmico de espectro (DSM) tem sido propostas para mitigar os efeitos do *crosstalk*. As soluções DSM otimizam a transmissão coordenando a potência transmitida pelos usuários levando em consideração os efeitos dessa potência na geração de *crosstalk* na rede [3–5].

Os trabalhos acadêmicos em DSM focaram-se inicialmente na criação de algoritmos que pudessem otimizar a utilização da faixa de espectro alocada a estes serviços, levando em consideração a natureza efetivamente imutável das condições de canal em cabos telefônicos. O uso de técnicas de DSM para maximização de taxa de bits foram exploradas em, por

exemplo, [6–11], enquanto que a utilização de DSM para minimizar o consumo de potência tem ganhado mais atenção em trabalhos recentes como: [12–14]. Trabalhos recentes também tem desenvolvido algoritmos que objetivam aumentar a estabilidade nas linhas de assinante, atualizando os valores de margem de forma a minimizar a ocorrência de períodos de indisponibilidade do serviço [15–18].

Uma outra corrente de pesquisa tem como objeto de estudo a implementação prática de algoritmos e técnicas de DSM em equipamentos reais, com e sem modificações. Em [19] os autores afirmam ter implementado o algoritmo IWF (*Iterative Waterfilling* [6]) no *firmware* dos próprios CPEs (*costumer premises equipment*, em outras palavras os modems que ficam na casa do usuário).

Em [20] e [21] é utilizada uma abordagem onde dados relacionados à performance das linhas são armazenados e analisados posteriormente para servir de base em um processo contínuo de otimização.

Em [22] foram apresentados dois *softwares* que em conjunto são capazes de otimizar o desempenho de uma rede DSL, calculando e aplicando soluções de algoritmos DSM de forma centralizada em equipamentos comerciais sem nenhuma modificação. Entretanto, as ferramentas apresentados em [22] por si só não são adequadas para uma possível implementação de DSM em redes comerciais. A otimização realizada no referido trabalho não leva em consideração, por exemplo, a presença de tráfego útil nos enlaces.

Esta dissertação descreve a implementação de um protótipo para implementação e aplicação de estratégias de otimização em sistemas DSL. Busca-se com tal protótipo poder aplicar os diversos algoritmos DSM implementados em [22] de uma rede DSL real, com equipamentos comerciais não modificados.

Ao contrário da otimização pontual aplicada em [22], busca-se com este protótipo a possibilidade de otimização contínua e em tempo real, em uma rede DSL com fluxos de tráfego dinâmicos. Múltiplas linhas em uma rede DSL são monitoradas, e o sistema decide, de forma autônoma, não só como mas também quando aplicar medidas de otimização evitando interromper conexões ativas. Além disso, o protótipo desenvolvido agrega uma série de funcionalidades para controle e obtenção de parâmetros de equipamentos DSL, possibilitando a implementação e avaliação de novas estratégias. A seguir são descritos os objetivos dessa dissertação.

## 1.2 Objetivos

O presente trabalho baseia-se no conceito de otimização DSM centralizada aplicada à equipamento DSL não modificado desenvolvido em [22]. Os principais objetivos deste trabalho são os seguintes:

- Desenvolvimento de um protótipo para teste e validação de técnicas de gerenciamento dinâmico de espectro que baseiam-se no monitoramento contínuo do estado das linhas e em múltiplas variáveis de interesse (taxa de bits, potência utilizada, margem, atividade do enlace, entre outras). O protótipo constitui-se de um *software* para monitoramento, tomada de decisões, execução e aplicação dos resultados de algoritmos DSM, além de um mecanismo eficiente para geração de estímulos de tráfego e ruídos em um número considerável de linhas.
- Utilizando o protótipo desenvolvido, discutir a implementação e resultados da aplicação de uma estratégia de otimização nível 1 chamada AMA (*Automatic Margin Adaptation*) em um cenário prático com onze usuários.
- Também valendo-se do protótipo, apresentar resultados práticos comparativos do desempenho de um algoritmo DSM nível 2 (ISB - *Iterative Spectrum Balancing* [7, 23]), utilizando informação de canais de *crosstalk* provenientes de duas fontes: medidas e um método de estimação descrito em [24].

## 1.3 Organização do Trabalho

Este trabalho está organizado da seguinte forma:

- No Capítulo 2 são revistos alguns dos fundamentos da tecnologia DSL, bem como são descritos os principais algoritmos DSM e sua relevância para este trabalho.
- No Capítulo 3 é apresentado o projeto e implementação do protótipo para monitoramento e otimização contínua de sistemas DSL.
- Em seguida, no Capítulo 4 são descritos os resultados da aplicação do protótipo em duas aplicações teste, descritas na seção anterior.
- Finalmente, o Capítulo 5 traz as considerações finais a respeito do trabalho desenvolvido.

# Capítulo 2

# Sistemas xDSL, margem e DSM

Este capítulo faz um apanhado sucinto sobre as principais tecnologias DSL encontradas no mercado, além de apresentar os principais tipos de interferência nestes sistemas e introduzir o conceito de margem. Também são apresentados aqui o modelo do sistema adotado neste trabalho e uma discussão sobre técnicas de gerenciamento dinâmico de espectro.

## 2.1 Principais tecnologias DSL

As principais tecnologias DSL em operação hoje em dia são: ADSL, ADSL2, ADSL2+, VDSL e VDSL2. O desenvolvimento e padronização dessas tecnologias são realizados principalmente pelo comitê ANSI (*American National Standards Institute*) T1E1.4 (*Digital Subscriber Loop Access*), pelo grupo de trabalho ETSI (*European Telecommunications Standards Institute*) TM6 (*Transmission and Multiplexing*), e pelo grupo de estudo ITU-T 15/Question 4, bem como também por organizações industriais, tais como o DSL Forum [25].

A Tabela 2.1 lista os principais serviços e os respectivos documentos de padronização.

## 2.2 Fontes de ruído e interferência

A transmissão de dados usando par trançado é sujeita a várias formas de interferências, apresentando barreiras que podem chegar a inviabilizar o fornecimento do serviço DSL. Em [32] essas fontes de ruídos são classificadas em dois grupos: *limitante de qualidade* e *limitante de capacidade*. Nas Seções 2.2.1 e 2.2.2 serão descritos com mais detalhes cada um desses tipos de interferência.

Tabela 2.1: Principais serviços DSL e os respectivos documentos de padronização.

| Serviço | Padrão | Referências |
|---|---|---|
| ADSL | ITU-T G.992.1, ANSI T1.413 Issue 2 | [26], [27] |
| ADSL2 | ITU-T G.992.3 | [28] |
| ADSL2+ | ITU-T G.992.5 | [29] |
| VDSL | ITU-T G.993.1 | [30] |
| VDSL2 | ITU-T G.993.2 | [31] |

### 2.2.1 Limitante de qualidade

Os ruídos que limitam a qualidade de sistemas DSL (em taxa de erro de bits) são considerados ruídos extrínsecos ao par trançado. Um deles é o ruído impulsivo, originado em descargas elétricas, lâmpadas fluorescentes, acionamento de motores, cercas eletrificadas e etc. Existem também interferências recebidas de rádios amadores, chamadas de RFI (*Radio Frequency Interference*).

Esse tipo de interferência é imprevisível e variável geograficamente, e são levados em conta em projeto de sistemas DSL através de uma margem de segurança, que basicamente reduz a capacidade do canal, mas aumenta a estabilidade da linha (mais detalhes sobre o funcionamento da margem na Seção 2.4). Além disso, algumas outras técnicas são usadas para combater esse tipo de ruído, como técnicas de codificação de erro. Exemplo: FEC (*Forward Error Correction*) para detecção e correção de erros no receptor; e *interleaving*, para combater erros em rajada [2].

### 2.2.2 Limitante de capacidade

Algumas das interferências são consideradas intrínsecas ao ambiente de par trançados e de certa forma podem ser preditas. Exemplos: ruído térmico, ecos e reflexões, atenuação e *crosstalk*.

#### 2.2.2.1 O *crosstalk*

O *crosstalk* é o acoplamento eletromagnético que ocorre entre condutores que estão em um mesmo *binder*[1], e é a principal causa da degradação da capacidade (em bits por

[1]Um *binder* é um conjunto de pares trançados dentro de um cabo telefônico.

segundo) de sistemas DSL. Os padrões DSL procuram combater esse mal propondo modelos de compatibilidade espectral, que basicamente definem máscaras de transmissão máxima de modo a facilitar a predição da quantidade de interferência que os receptores devem lhe dar.

Existem dois tipos de *crosstalk*: o *near-end crosstalk* (NEXT) e o *far-end crosstalk* (FEXT). O NEXT é a interferência proveniente de um transmissor que afeta a recepção de outro usuário na mesma extremidade do cabo. O FEXT, por sua vez, afeta a recepção de um usuário na extremidade oposta da linha.

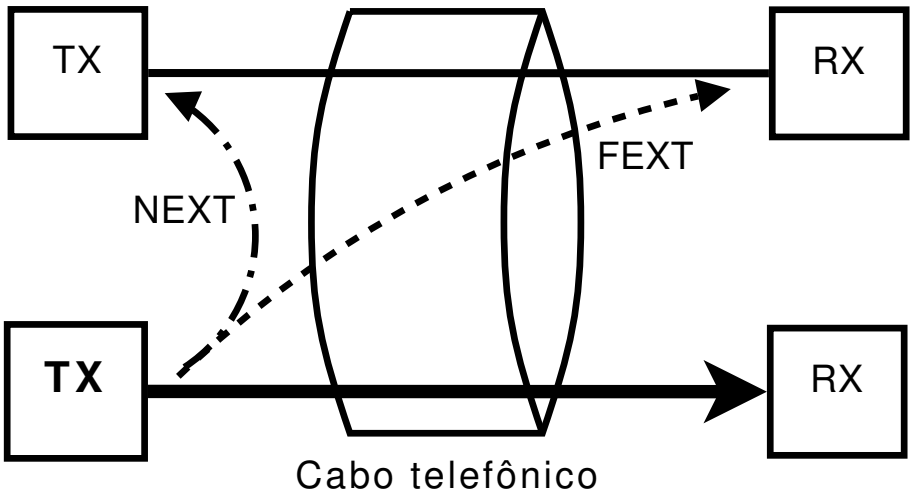


Figura 2.1: Tipos de *crosstalk*.

A intensidade da interferência de NEXT é bem maior que a de FEXT, visto que esse último é atenuado até atingir o receptor da outra ponta. As tecnologias atuais de DSL evitam o NEXT utilizando uma técnica é conhecida como *frequency division duplex* (FDD). Por esse motivo, costuma-se desprezar as interferências de NEXT, considerando apenas o FEXT como principal fonte de interferência.

## 2.3 Modelo do sistema

Os sistemas DSL considerados neste trabalho podem ser modelados como um *binder* consistindo de $N$ usuários ($N$ linhas) equipados com transmissores DSL. Cada transmissor emprega a modulação *discrete multitone* (DMT) e opera sobre um par trançado com $K$ subcanais paralelos (tons), que são livres de interferência intersimbólica [1]. Assumindo o uso de duplexação por divisão de frequência (FDD) na modulação DMT (bandas de *upstream* e *downstream* não sobrepostas), somente o *far-end crosstalk* (FEXT) é considerado. A fraca

influência do *crosstalk near-end* (NEXT) é desconsiderada [1].

O problema de alocação de bits (*bit loading*) para o tom $k$ e usuário $n$ pode ser expresso como descrito na Equação 2.1,

$$b_n^k = \log_2\left(1 + \frac{SNR_n^k}{\tilde{\gamma}_u \Gamma}\right) \tag{2.1}$$

onde:

- $b_n^k$ é o número de bits alocados no tom $k$ para o usuário $n$.
- $SNR_n^k$ é a razão sinal-ruído (*signal-to-noise ratio*) no tom $k$ para o usuário $n$.
- $\tilde{\gamma}_u$ é a margem (ou SNR *margin*. Ver Seção 2.4).
- $\Gamma$ Denota o *gap* da razão sinal ruído, que é uma função da taxa de erro de bits (BER) desejada, tipicamente $10^{-7}$. O *gap* é um indicador de quão próximo a taxa de bits está em relação à capacidade teórica do canal. [1].

A SNR para o tom $k$ e usuário $n$ é definida conforme a Equação 2.2,

$$SNR_n^k = \frac{s_n^k \left|h_{n,n}^k\right|^2}{\left(\sigma_n^k + \sum_{m \neq n} s_m^k \left|h_{n,m}^k\right|^2\right)} \tag{2.2}$$

onde:

- $s_n^k$ denota a potência alocada para o usuário $n$ no tom $k$.
- $|h_{n,n}^k|^2$ denota o quadrado da magnitude do ganho do canal direto para o usuário $n$ no tom $k$.
- $\sigma_n^k$ representa a potência do ruído de fundo no tom $k$ no receptor do tom $n$.
- $|h_{n,m}^k|^2$ denota o quadrado da magnitude do canal de FEXT do transmissor $m$ para o receptor $n$ no tom $k$.

## 2.4 Margem

Para lidar com as condições variantes do ruído em linhas digitais de assinante, os padrões xDSL adotam a utilização de um parâmetro chamado margem. Em poucas palavras, o conceito de margem pode ser enunciado como: "quanto a SNR pode ser degradada sem que os modems precisem refazer a alocação de bits" [33].

Em termos práticos, a margem é implementada reservando uma porção da SNR que não será utilizada para alocar bits. A Fig. 2.2 traz um exemplo gráfico. A curva em preto é a SNR original. A curva em vermelho é a SNR efetivamente utilizada para alocação de bits. Na situação retratada na figura, a potência do ruído em qualquer um dos tons pode aumentar até 9 dB sem que ocorra degradação da taxa de erro de bits no receptor.

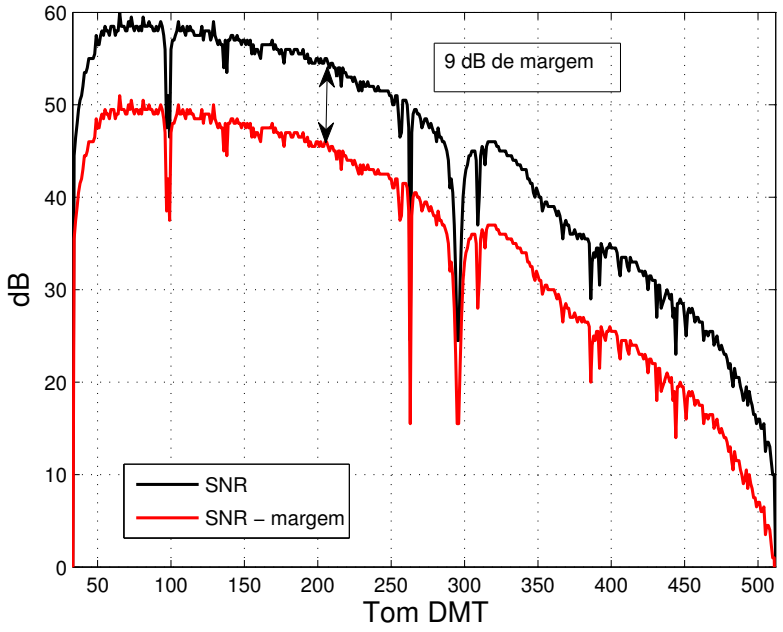


Figura 2.2: Ilustração da utilização de margem em sistemas DSL. A diferença entre as curvas é o valor da margem. Os valores de SNR retratados no gráfico foram obtidos de um modem comercial em laboratório.

Em equipamentos xDSL comerciais é possível especificar o valor de margem a ser utilizado pelos modems. Entretanto, ao invés de um valor de margem por tom, utiliza-se um único valor chamado **margem alvo** (TSNRM [28]). A margem alvo é utilizada pelos modems durante a inicialização, ao executar a alocação de bits. Os receptores realizam a alocação de bits subtraindo a margem alvo da SNR, como mostrado na Fig. 2.2.

Após a inicialização as condições de ruído podem mudar, e os modems não podem garantir que a taxa alvo seja mantida igual em todos os subcanais. O equipamentos xDSL comerciais informam, através do parâmetro SNR *margin*, o valor "instantâneo" da margem. Semelhante ao que acontece com a margem alvo, a SNR *margin* também é informada como um valor escalar, que equivale a média dos valores de margem instantânea por tom.

Neste trabalho, adota-se a seguinte convenção:

- O valor de margem utilizado na inicialização, para calcular a alocação de bits é chamado **margem alvo**.
- O valor de margem média que varia com o tempo devido às condições de ruído é chamado de **margem** ou **SNR *margin***.

## 2.5 Algoritmos DSM

Como explicado na Seção 2.2, o *crosstalk* limita bastante a capacidade das redes DSL. De forma a diminuir essa perda, os sistemas DSL atuais vem sendo projetados para operar sempre no "pior caso" de *crosstalk*, levando a desempenhos bastante conservadores. Esse modo de operação é usualmente chamado de gerenciamento estático de espectro (*static spectrum management* - SSM), e é o que utilizamos comercialmente nos dias atuais.

Em 2002, surgiu o conceito de gerenciamento dinâmico de espectro (*dynamic spectrum management* - DSM), no qual os aspectos multiusuários do ambiente DSL são levados em conta na transmissão [5].

### 2.5.1 Níveis de coordenação de espectro

As técnicas de DSM geralmente são classificadas nas seguintes categorias, de acordo com o nível de coordenação necessário entre as linhas [34]:

- DSM nível 0: corresponde ao gerenciamento estático do espectro (SSM), o que, na prática, significa a ausência de qualquer algoritmo DSM.
- DSM nível 1: As linhas são otimizadas individualmente, sem considerar interação entre os usuários. Encaixam-se nessa categoria tanto algoritmos de balanceamento de espectro, como o IWF [6], quanto algoritmos que fazem ajustes de parâmetros de configuração como o AMA (*Automatic Margin Adaptation*).

- DSM nível 2: Altera as alocações de potência dos usuários, considerando a interação entre eles. Nesse caso, o uso de um agente central se faz necessário, de forma a coordenar a alocação de potência dos usuários, minimizando o impacto do *crosstalk*.[2]

- DSM nível 3: também conhecido como *Vectoring* (transmissão vetorizada), no qual os sinais de todas as linhas são transmitidos de forma conjunta, de forma a cancelar o *crosstalk* [35]. Não podem ser implementados em equipamento atual sem modificações.

Este trabalho trata somente de técnicas de reconfiguração e balanceamento de espectro (DSM níveis 1 e 2), que podem ser implementados em *hardware* comercial já existente. Os algoritmos de nível 3 não são considerados.

## 2.6 Estados de um transmissor xDSL

Este trabalho faz menção a alguns termos sobre estados [28] que um transmissor DSL pode assumir. São eles:

- ***showtime*** - É o estado alcançado ao fim do processo de inicialização, em que os modems podem efetivamente transmitir dados. Por exemplo, quando estão sendo utilizados para navegar na *internet* os modems DSL estão em *showtime*.

  Os termos ***up*** e ***down*** podem ser utilizados para identificar o estado de uma linha. A palavra *up* indica que a linha está em *showtime*. O termo *down* indica que a linha se encontra fora do estado de *showtime*.

- **retreino** - Um retreino acontece quando um receptor DSL ultrapassa a BER limite. O receptor sai do estado de *showtime* e vai para o estado ***init/train***, executando, entre outras operações, a alocação de bits. Durante o retreino, o tráfego de dados no enlace é interrompido.

[2] Agente central: pode ser um Centro de Gerenciamento de Espectro (*Spectrum Management Center* - SMC) ou um Multiplexador de Acesso DSL (*Digital Subscriber Lines Access Multiplexer* - DSLAM) para DSM centralizado [34].

# Capítulo 3

# Um Protótipo para Monitoramento e Otimização Contínua de Sistemas DSL

## 3.1 Introdução

Este capítulo descreve o projeto e implementação de um protótipo para monitoramento e otimização contínua de sistemas DSL. O referido protótipo constitui-se de um *software*, descrito na Seção 3.2 e *setups*[1] experimentais para geração de tráfego e geração de ruído, descritos na Seção 3.3 que possibilitam reproduzir em laboratório condições semelhantes às encontradas em sistemas DSL instalados em uma área urbana.

## 3.2 O *software* DLM

Esta seção descreve o projeto e implementação de um *software*, chamado DLM (*Dynamic Link Management*), que executa otimização contínua em redes DSL, de forma que as linhas possam alcançar os níveis de desempenho desejados. Do ponto de vista de uma operadora, o *software* proposto pode minimizar problemas de instabilidade, diminuir os níveis de crosstalk entre os usuários gerenciados e possibilitar a venda de mais serviços, aumentando o valor agregado por usuário. Sob outra perspectiva, a aplicação de DSM através do software proposto pode ainda reduzir os custos operacionais, através da diminuição do número de intervenções em campo devido à linhas que sofrem retreinos frequentemente (linhas instáveis).

O *software* DLM compartilha algumas funcionalidades com trabalhos previamente

[1] Um conjunto de equipamentos e suas respectivas configurações.

citados [20, 21], mas diferencia-se por seu *design*. Ele é focado em prover diferentes políticas de otimização, de forma que uma operadora tenha liberdade para escolher como otimizar sua rede de acordo com os diferentes serviços oferecidos (acesso à internet, voz e vídeo-sobre-IP entre outros). Além disso, a interação entre o DLM e os DSLAMs (*Digital Subscriber Line Access Multiplexer*) é totalmente baseada em comandos do protocolo SNMP(*Simple Network Management Protocol*) [36]. O código do DLM é executado em uma máquina à parte, de forma que os recursos reduzidos de memória e processamento em um DSLAM não sejam diretamente afetados por sua execução. Na implementação do software, utiliza-se o paradigma de orientação a objetos, através da linguagem Java.

O *software* descrito aqui pode ser considerado uma extensão do exposto em [22]. Em contraste com a otimização pontual descrita no trabalho anterior, aqui as técnicas de DSM são aplicadas de forma contínua e não-invasiva, proporcionando uma visão mais completa sobre os ganhos possíveis na vazão (*throughput*), consumo de energia e estabilidade de um sistema DSL, diminuindo as distâncias para uma futura aplicação comercial de gerenciamento dinâmico em sistemas DSL.

Na próxima seção, uma visão geral das funcionalidades do software é apresentada.

### 3.2.1 Uma visão geral das funcionalidades

O DLM é um *software* responsável por otimizar a transmissão em redes DSL, baseando-se na interação entre as diferentes linhas. Para tanto, o estado de cada linha é monitorado, e o *software* pode decidir por aplicar algoritmos DSM ou fazer um ajuste fino de parâmetros de tal forma que cada linha cumpra seus requisitos de taxa de bits e estabilidade.

O software proposto é baseado na idéia de diferentes *estratégias de otimização*. Uma estratégia define como e quando técnicas de DSM (nível 1 ou 2) e o ajuste de parâmetros de configuração devem ser usados para otimizar o desempenho de cada usuários gerenciado.

Cada estratégia recebe *eventos* como entrada e os utiliza para decidir que *ações* tomar. Eventos são modelados a partir de dados obtidos dos DSLAMs em intervalos regulares de tempo. Podem representar, por exemplo, instabilidade em uma determinada linha. Por sua vez, as *ações* representam comandos definidos pelas estratégias, por exemplo, mudança na taxa alvo de determinado usuário, ou uma ordem para calcular uma nova máscara de PSD.

O DLM assume a existência de um conjunto de linhas ativas, cada uma com seus requisitos de taxa de bits e estabilidade previamente definidos.

A próxima seção descreve a arquitetura do *software*. Nas seções subsequentes, cada um dos blocos funcionais expostos na Fig. 3.1 é descrito.

### 3.2.2 Blocos funcionais

Os principais componentes da solução proposta e suas relações são apresentados na Fig. 3.1. Os números entre parênteses na figura são utilizados no texto a seguir para identificar cada bloco.

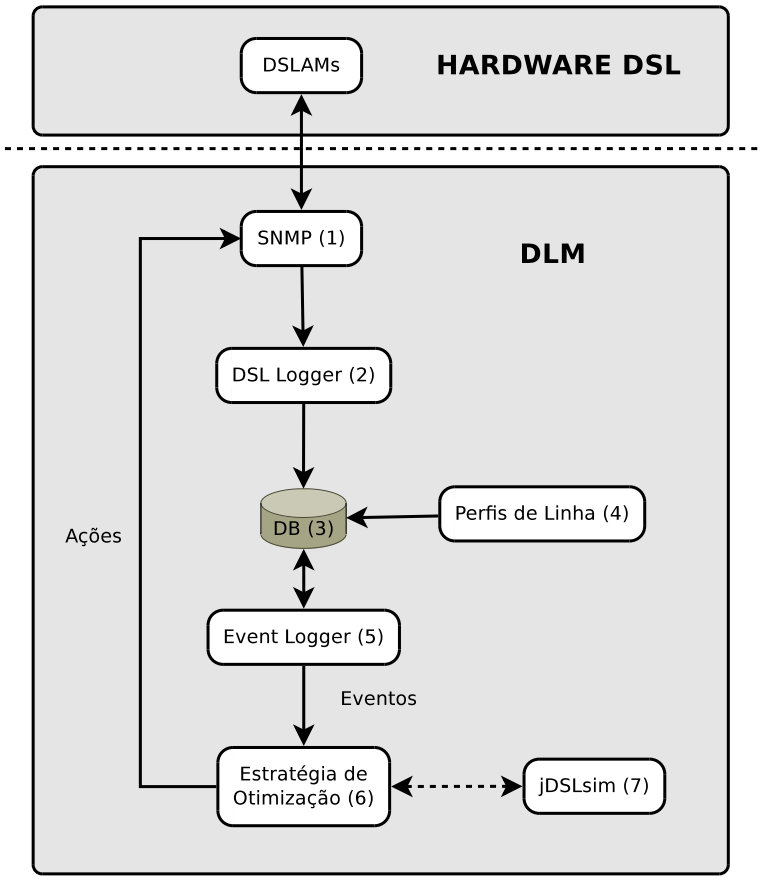


Figura 3.1: Principais blocos funcionais do DLM. As flechas indicam o fluxo dos dados. Linhas pontilhadas indicam caminhos opcionais.

- No nível mais baixo (em contato direto com o *hardware* DSL), o bloco *SNMP* **(1)** é responsável por transmitir mensagens de controle e receber respostas dos DSLAMs. É importante diferenciar o *bloco* SNMP do *protocolo* SNMP em si. No contexto deste trabalho, o termo "bloco SNMP" descreve um subsistema do software DLM que implementa funcionalidades do protocolo SNMP, e é utilizado para controlar equipamentos DSL.

- O bloco SNMP é principalmente acessado pelo bloco *DSL Logger* **(2)** para executar consultas relacionados às condições das linhas em um determinado momento. O DSL

Logger envia as informações coletadas para um banco de dados **(3)**.

Os parâmetros monitorados por padrão são: taxa de bits, margem, estado da linha (*up* / *down*), potência de saída e o nível de utilização do enlace, obtido através da contagem de células ATM (*Asynchronous Transfer Mode*) que carregam carga útil.

- O banco de dados **(3)** recebe informações e consultas dos blocos DSL **(2)** e *Event Logger* **(5)**. Também armazena os perfis **(4)** (conjunto de configurações) para cada linha.

- O bloco *Event Logger* **(5)** consulta o banco de dados para identificar *eventos* de interesse. Instabilidade em um determinado usuário, por exemplo, pode ser detectada através da contagem do número de retreinos durante um determinado período de tempo.

  O Event Logger constrói um histórico que contém os eventos para cada linha e o armazena no banco de dados. Eventos são as entradas principais para as *estratégias de otimização* **(6)**.

  Os eventos registrados por padrão são: retreino, inatividade (ausência de tráfego IP relevante no enlace) e falha em atender os requisitos.

- *Estratégias de otimização* **(6)** decidem que *ação* tomar baseando-se em *eventos* discretos. Uma estratégia define, por exemplo, quais linhas devem ser otimizadas, que parâmetros devem ser modificados, e se uma nova limitação de potência deve ser imposta. Estratégias que empregam algoritmos DSM podem utilizar uma instância do software jDSLsim **(7)** [22] para executar o balanceamento de espectro.

  As estratégias executam comandos através de *ações*. que podem ser descritas por sentenças simples como: "aumentar a margem alvo na linha $n$". Ações são traduzidas em comandos SNMP pelo bloco SNMP **(1)**.

Nas seções a seguir, cada um dos blocos numerados será descrito em mais detalhes, de acordo com a ordem definida na Fig 3.1.

### 3.2.3 O bloco SNMP

Os DSLAMs atuais expõe através do protocolo SNMP (*Simple Network Management Protocol*) [36] uma série de controles e parâmetros que podem ser utilizados pelo operador para monitorar e controlar cada uma das linhas em sua rede. Taxa de bits, potência de saída, margem, alocação de bits, SNR por tom, são alguns dos parâmetros disponíveis, identificados por códigos únicos chamados OIDs (*Object Identifiers*).

Cada OID identifica um único parâmetro e pode ser utilizado para recuperar seu respectivo valor em um determinado instante. As consultas SNMP são feitas diretamente ao DSLAM, através da rede.

O *software* DLM emprega o protocolo SNMP para realizar todas as interações com os DSLAMs, desde o monitoramento (DSL Logger), execução comandos para alterar o estado das linhas e configuração de parâmetros. O subsistema responsável chama-se SNMP e traduz comandos de alto nível em consultas SNMP que os DSLAMs possam entender, também realizando as conversões no sentido contrário.

Como o DLM foi projetado para gerenciar por volta de cem linhas, seria inviável enviar uma requisição SNMP para cada um dos parâmetros sendo monitorados em cada linha. O protocolo SNMP provê uma forma de agrupar vários identificadores de objeto em uma única requisição (*GetBulkRequest-PDU* [36]). O bloco SNMP do DLM baseia-se nessa característica, o que resultou em atrasos pequenos nas consultas, possibilitando que os valores para cada parâmetro possam ser registrados, por exemplo, a cada segundo.

A Fig. 3.2 representa os passos para a execução de uma requisição ao DSLAM utilizando o bloco SNMP do DLM. Seu elemento principal é a classe *SnmpQuery*. Este objeto armazena uma série de operações/comandos para serem executados sequencialmente pelo DSLAM, e é também responsável por retornar valores inteligíveis a partir das respostas fornecidas (os valores retornados pelos DSLAMs são codificadas de forma específica para cada tipo de OID. Valores de margem, por exemplo, são codificados como inteiros onde cada unidade representa 0.1 dB. Valores de PSD são codificados como *strings* hexadecimais. Faz-se necessário um procedimento de conversão explícita para cada parâmetro individualmente).

A sequência de operação apresentada na Fig. 3.2 é descrita abaixo para uma requisição do tipo *get* (leitura do valor de um determinado OID):

- Inicialmente, um objeto *SnmpQuery* é criado.

```
SnmpQuery query = new SnmpQuery(ipAddress, port, dslamModel);
```

- A seguir, os comandos de interesse são registrados (agendados para serem executados sequencialmente, quando a consulta SNMP for transmitida pela rede).

```
// agendamento de parâmetros a serem consultados
query.registerGetAdminStatus(line);
```

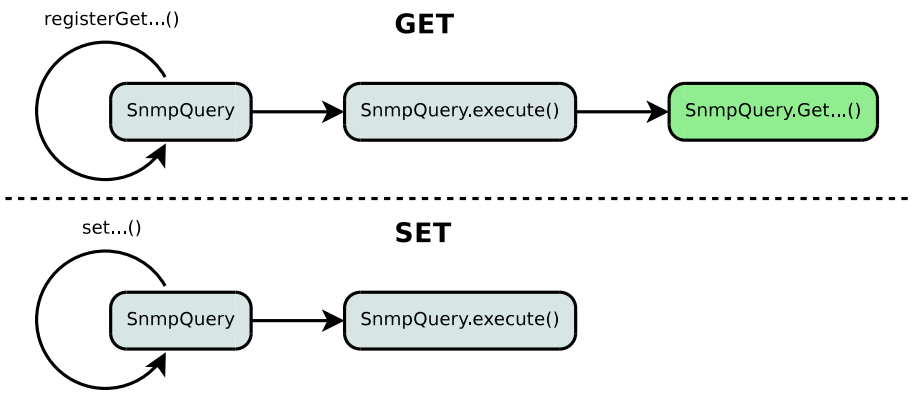


Figura 3.2: Fluxograma da execução de requisições SNMP do tipo *get* (leitura) e *set* (escrita) [36] no software DLM. As setas em *loop* em volta da classe SnmpQuery indicam que vários parâmetros podem ser registrados antes que as requisições sejam executadas. O bloco em verde representa as funções que retornam valores decodificados (convertidos para uma escala significativa).

```
query.registerGetMaxSnrMgnDS(line);
```

- Depois de todos os comandos serem registrados, a requisição deve ser executada. Nesse ponto, o bloco SNMP envia um pacote SNMP pela rede e recebe as respostas do DSLAM.

```
// envia pacotes SNMP
query.execute();
```

- Para receber as respostas convertidas às suas devidas escalas, o cliente (um trecho de código que utilize a classe *SnmpQuery*) então chama as funções que registrou anteriormente.

```
// retorna o estado de uma linha
x = query.getAdminStatus(line);

// retorna o valor de margem downstream
y = query.getSnrMgnDS(line);
```

Um ponto forte da implementação escolhida para o bloco SNMP do DLM é que além de reduzir a latência agrupando comandos, os objetos *SnmpQuery* podem ser armazenados para execução periódica. Em outras palavras, uma vez que um objeto *SnmpQuery* tiver registrado todos os comandos de interesse ele pode ser executado repetidamente, uma característica importante para a aplicação, que monitora periodicamente um grupo de parâmetros pré-definido por longos períodos.

Um outro ponto a ressaltar no bloco SNMP do DLM é que ao criar um objeto *SnmpQuery* é preciso especificar o modelo do DSLAM que será consultado. Isso possibilita a utilização do *software* com equipamentos diferentes, já que o bloco SNMP está preparado para tratar, por exemplo, o caso em que um mesmo parâmetro é identificado por OIDs diferentes em equipamentos diversos. O bloco SNMP é independente do resto do *software* DLM, podendo ser utilizado por outras aplicações para interfacear com DSLAMs. No Capítulo 4 é apresentada uma aplicação desse pacote na implementação de um estimador para funções de transferência de canais de *crosstalk*.

Em seguida é descrito o bloco funcional do DLM responsável por monitorar periodicamente o desempenho das linhas digitais de assinante.

### 3.2.4 DSL Logger

O subsistema DSL Logger (bloco 2 na Fig. 3.1) é responsável por executar consultas periódicas para monitorar o desempenho das linhas. Os parâmetros monitorados são:

- Taxa de bits
- Margem atual
- Estado da linha (*up* / *down*)
- Potência de saída
- Nível de utilização do enlace

O nível de utilização no enlace é calculado de acordo com a Eq. 3.1, onde $L_{act}$ representa o nível de atividade no enlace, $N_{idle}$ é o número de células ATM vazias e $N_{total}$ é o número de células ATM transmitidas naquele canal em um intervalo de 15 minutos.

$$L_{act} = 1 - (N_{idle}/N_{total}) \tag{3.1}$$

A Tabela 3.1 contém um exemplo dos dados brutos coletados durante a execução do DSL Logger para uma única linha (usuário) em laboratório. Tabelas semelhantes, armazenadas no banco de dados são utilizadas para detecção de eventos que podem definir e disparar o processo de otimização (mais detalhes na Seção 3.2.6).

Tabela 3.1: Saída exemplo do DSL Logger para uma única linha no sentido *downstream*. OUTPWR é a potência de saída atual. LINESTATE indica se uma linha está ou não transmitindo (*showtime*). SNRM é o valor de margem. BITRATE é a taxa de bits. LINE_ACT é um indicador da quantidade de tráfego útil sendo transmitido nessa linha, calculado através de uma relação entre o número de células ATM que carregam alguma carga útil e o número total de células ATM transmitidas nesse enlace.

| N | TIMESTAMP | OUTPWR | LINESTATE | SNRM | BITRATE | LINE_ACT |
|---|---|---|---|---|---|---|
| 1 | 19:01:45 | 0 | down | 0 | 0 | 0 |
| 2 | 19:02:00 | 0 | down | 0 | 0 | 0 |
| 3 | 19:02:15 | 14.6 | up | 6.1 | 14671 | 0 |
| 4 | 19:02:29 | 14.6 | up | 6.1 | 14671 | 0 |
| 5 | 19:02:44 | 0 | down | 0 | 0 | 0 |
| 6 | 19:03:00 | 0 | down | 0 | 0 | 0 |
| 7 | 19:03:14 | 19.4 | up | 6.2 | 12610 | 0.13 |
| 8 | 19:03:29 | 19.4 | up | 6.2 | 12610 | 0.15 |

O DSL Logger emprega o bloco SNMP apresentado na seção anterior e se beneficia do comportamento da classe *SnmpQuery*. Inicialmente, um objeto é criado e os parâmetros supracitados são registrados. Esse objeto é utilizado enquanto o DLM estiver sendo executado para executar consultas em intervalos de tempo regulares.

A execução periódica de tarefas é baseada nas classes do pacote *java.util.concurrent* [37], fornecido na biblioteca padrão das versões mais recentes da plataforma Java. O DSL Logger faz uso de classes desse pacote para executar tarefas periodicamente, ao invés da antiga abordagem de utilizar múltiplas *threads* e executar um controle fino sobre os estados de cada fluxo de execução.

O projeto deste bloco é flexível o suficiente para que os parâmetros a serem registrados sejam escolhidos pelo cliente. Assim como o bloco SNMP, o DSL Logger pode ser utilizado de forma independente da aplicação principal (DLM), em tarefas que necessitem registrar dados dos DSLAMs ao longo do tempo.

### 3.2.5 Banco de dados

O bloco de banco de dados do DLM ( bloco 3 na Fig. 3.1 ) é responsável por armazenar informação do estado das linhas, eventos, parâmetros de configuração para cada DSLAM além dos perfis de linha. Os dados registrados pelo DSL Logger (descrito na última seção) são armazenados pelo banco de dados. Os blocos Event Logger e as Estratégias de Otimização também consultam o banco respectivamente para detectar eventos e tomar decisões.

O banco de dados foi implementado de tal forma a tornar a persistência de informações transparente. Em outras palavras, comandos SQL (*Structured Query Language*) foram substituídos por uma camada de persistência baseada em JPA (*Java Persistence API* [38]). Essa API (*Application Programming Interface*) é responsável por mapear objetos Java para um banco de dados relacional (e vice-versa) automaticamente, sem que o desenvolvedor precise escrever código SQL. Como resultado dessa escolha de projeto, o bloco de banco de dados do DLM é independente do sistema de gerenciamento de banco de dados, do provedor de persistência [38] e da sintaxe SQL. Mudanças na estrutura das classes persistidas não geram a necessidade de reescrita de código SQL.

### 3.2.6 O bloco Event Logger

Apesar da informação gerada pelo DSL Logger poder ser utilizada para avaliar a situação de cada uma das linhas, seria inconveniente implementar estratégias de otimização baseadas nos valores brutos dos parâmetros. Ao invés disso, esses valores são filtrados de forma a identificar "eventos" discretos que podem ser usados pelas estratégias para tomada de decisões.

Como exemplo prático do que vem a ser um evento, considere as linhas de 4 a 7 na Tabela 3.1. É possível notar que inicialmente, a linha está em *showtime* (identificado pelo valor "*up*" na coluna LINESTATE). Nas próximas duas linhas, LINESTATE foi reportado como "*down*" e então na linha 7, volta para "*up*". Esta sequência de transições é característica de uma linha que passou por um retreino, geralmente devido à incapacidade de lidar com a piora nas condições de ruído no *binder* [28].

O bloco Event Logger (ver Fig. 3.1) identifica de forma autônoma a ocorrência de tal *Evento de Retreino* e armazena a informação relacionada no banco de dados. Desta forma, as estratégias de otimização não precisam verificar as informações de estado das linhas frequentemente, focando-se, ao invés disso em fazer consultas ao banco de dados para obter eventos aos quais devem reagir.

O Event Logger não interage com os DSLAMs nem gera tráfego SNMP. É executado periodicamente em uma *thread* independente, consultando o banco de dados e armazenando sua saída também no banco, em uma tabela específica (tabela EVENTS). Até a presente data, o bloco Event Logger detecta três condições especiais: retreinos, inatividade no enlace (muito poucos pacotes sendo enviados) e falha em atender aos requisitos. As condições para detectar tais eventos, bem como sua importância e aplicabilidade são descritos à seguir.

#### 3.2.6.1 Inatividade no enlace

A maioria dos parâmetros de configuração que podem ser utilizados para otimizar o desempenho em sistemas DSL não podem ser alterados em *showtime*. Este fato impõe uma dificuldade para algoritmos DSM de nível 1 e 2 que precisam alterar, por exemplo, margens alvo ou máscaras PSD. O DLM tenta minimizar o impacto desses processos de otimização sobre a experiência do usuário, determinando quando uma certa linha está ativa (transmitindo informações úteis) e evitando tomar qualquer medida que interrompa o serviço se esta condição for verdadeira.

A Tabela 3.1 contém uma coluna chamada "LINE_ACT" (*Nível de atividade no enlace*) que indica a quantidade de tráfego ATM relevante em um determinado canal. Cada estratégia de otimização implementada pode definir um limiar de atividade do enlace para determinar quando uma linha pode ser escolhida para otimização sem prejudicar a experiência do usuário. Quando detecta-se que uma linha está abaixo desse limiar, um evento de inatividade (INACTIVE_LINE *event*) é registrado.

A Tabela 3.2 lista os eventos descobertos pelo bloco Event Logger baseando-se nas informações de estado de linha contidos na Tabela 3.1. As duas tabelas são relacionadas pelas marcações de tempo presentes em ambas (coluna TIMESTAMP).

Considerando a coluna LINE_ACT na Tabela 3.1, é possível notar que o valor reportado é 0 durante o intervalo [19:01:45 - 19:03:00]. As linhas $1-5$ e 7 na Tabela 3.2 contém eventos de inatividade (LINE_INACTIVE) correspondendo a este intervalo. É possível também notar que de 19:03:14 em diante (linhas 7, 8 na Tabela 3.1), o valor da coluna LINE_ACT é diferente de 0, indicando a presença de tráfego gerado pelo usuário no enlace.

#### 3.2.6.2 Retreinos

Mudanças nas condições de ruído no *binder* podem fazer as linhas retreinarem ao longo do dia. Conforme o DLM monitora o estado das linhas (ver coluna LINESTATE na Tabela 3.1), ele pode determinar se uma linha saiu do estado de *showtime* e registra

Tabela 3.2: Eventos detectados pelo bloco Event Logger tendo como entrada os dados da Tabela 3.1. Eventos de inatividade de enlace (INACTIVE_LINE) estão presentes para cada momento em que muito pouco ou nenhum tráfego IP está sendo transmitido no canal. (Ver linhas 1 a 5 da Tabela 3.1).

| N | CODE | TIMESTAMP | TYPE |
|---|---|---|---|
| 1 | InactiveLineEvent | 19:01:45 | INACTIVE_LINE |
| 2 | InactiveLineEvent | 19:02:00 | INACTIVE_LINE |
| 3 | InactiveLineEvent | 19:02:15 | INACTIVE_LINE |
| 4 | InactiveLineEvent | 19:02:29 | INACTIVE_LINE |
| 5 | InactiveLineEvent | 19:02:44 | INACTIVE_LINE |
| 6 | RetrainingEvent | 19:02:44 | RETRAINING |
| 7 | InactiveLineEvent | 19:03:00 | INACTIVE_LINE |

um evento de retreino (RETRAINING *event*). O número de retreinos em um determinado período de tempo pode indicar que uma linha está instável, utilizando as configurações atuais. Estratégias de otimização podem então utilizar esta informação para escolher um novo perfil (conjunto de valores de configuração) para a linha. Um outro uso para este tipo de evento é, oportunisticamente, utilizar o momento dos retreinos para executar a otimização, enquanto as linhas estão fora de serviço.

Um evento de retreino é detectado quando uma linha passa pela seguinte sequência de estados: *up-down-up* sem nenhuma intervenção de comandos externos. A Fig 3.3 traz um diagrama que representa a sequência de transições necessárias à detecção deste tipo de evento. Para fins das estratégias de otimização, o caso onde, por exemplo, uma linha é desabilitada intencionalmente para alteração de parâmetros não é considerado um retreino.

Um exemplo de retreino pode ser observado comparando a Tabela 3.1 e a Tabela 3.2. Considere as linhas de 4 a 7 na Tabela 3.1: O estado da linha vai de *up* para *down* e novamente para *up*. A Tabela 3.2 registra então um evento de retreino às 19:02:44, o exato instante em que o estado da linha de interesse foi reportado como *down* pela primeira vez.

#### 3.2.6.3 Falha em atender aos requisitos

Os perfis de linha do DLM permitem ao operador especificar requisitos explícitos tais como taxa de bits mínima ou máxima potência de saída. O bloco Event

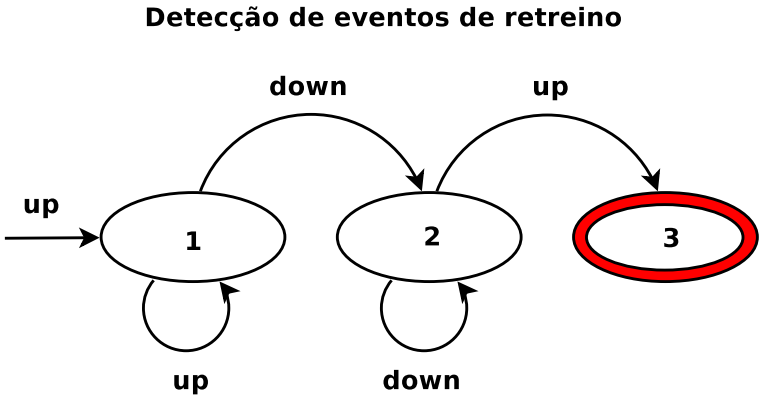


Figura 3.3: Máquina de estados representando o processo de detecção de um evento de retreino. As palavras que identificam as transições de estado são os valores do estado das linhas registrados em um determinado momento. O estado final (3), que indica detecção positiva está identificado em vermelho.

Logger é capaz de detectar caso uma determinada linha não consiga alcançar as restrições especificadas, disparando então um evento de falha em atender aos requisitos (REQUIREMENTS_NOT_MET *event*). Este tipo de evento pode ser utilizado pela estratégia de otimização para determinar que tipo de ação tomar para diferentes perfis de linha. Por exemplo, no caso em que uma linha não alcance sua taxa alvo, a mesma pode ser marcada como candidata a otimização de máscaras de PSD.

Detectar quais linhas falham em atender aos requisitos bem como quais estão inativas ou instáveis são aspectos chave para disparar e definir as ações que uma estratégia de otimização deve tomar para melhorar as condições de um sistema DSL. A próxima seção descreve as estratégias de otimização e como estas interagem com os blocos previamente descritos.

### 3.2.7 Estratégias de Otimização

Na estrutura do *software* DLM, as estratégias de otimização ocupam o nível mais alto (mais distante dos equipamentos DSL), sendo responsáveis pelo processo de tomada de decisões que visam otimizar o desempenho das linhas gerenciadas de acordo com algum critério.

Como afirmado anteriormente, o desenvolvimento de todo o *software* DLM objetiva possibilitar a criação e avaliação de estratégias de otimização diversas. Para tanto, buscou-se, na implementação desta camada, garantir flexibilidade e capacidade de extensão.

Tomando proveito dos conceitos de orientação a objetos presentes na linguagem de

implementação do DLM (Java), é fornecida uma implementação base de uma estratégia genérica. Esta estratégia genérica é executada em intervalos regulares de tempo, e tem à sua disposição (através de consultas ao banco de dados) o histórico dos eventos de interesse que ocorreram em todas as linhas gerenciadas. Esta implementação básica possui também meios de interagir com o *hardware* fazendo uso das funcionalidades providas pelo bloco SNMP. Utilizando herança, estratégias diversas podem ser implementadas utilizando técnicas DSM nível 1 ou nível 2 (contando com os vários algoritmos de balanceamento de espectro implementados no jDSLsim [22]).

Para a geração de parte dos resultados experimentais deste trabalho, reportados no Capítulo 4 implementou-se uma estratégia de nível 1 que executa controle automático das margens alvo. Uma descrição do funcionamento e implementação da mesma encontra-se na próxima seção.

#### 3.2.7.1 AMA - *Automatic Margin Adaptation*

Considerando o funcionamento de uma rede DSL comercial, com centenas de usuários, onde os cabos saem de uma central telefônica da operadora e passam por numerosas seções (emendas com diferentes bitolas, idades e grau de deterioração) até chegar ao lugar onde se encontram os equipamentos de usuário. Some-se a este quadro diversas fontes de ruído, como por exemplo rádios AM (interferência significativa na faixa de espectro dos sistemas ADSL e ADSL2+) e principalmente o crosstalk dos próprios usuários na rede. Constitui-se um ambiente hostil para o fornecimento de serviços como voz-sobre-IP e vídeo-sobre-IP, que são muito sensitivos à retreinos (durante um retreino, o serviço é interrompido). Em casos críticos, com linhas instáveis, até o tráfego de dados (hipertexto) torna-se inviável. Como resultado, as operadoras recebem chamadas de clientes insatisfeitos e precisam deslocar técnicos para verificar as condições de linha, um processo dispendioso e ineficiente.

Uma abordagem simplista para resolver o problema de instabilidade em sistemas DSL consistiria em configurar todas as linhas para que estas utilizassem o maior valor de margem possível (30 dB [28]). Esta solução tem sérios problemas, uma vez que, o aumento de margem alvo influi diretamente na forma como os transmissores executam a alocação de bits. Como visto anteriormente, a margem funciona como uma faixa da SNR não utilizada para alocação de bits. Para garantir margens maiores, os transmissores podem tomar duas medidas:

- Aumentar a potência transmitida, tendo como efeito colateral a geração de mais crosstalk na rede;
- Diminuir a faixa de SNR efetivamente utilizada para alocação de bits, o que resulta

em taxas de bits menores, e pode levar as linhas a não cumprirem seus requisitos de desempenho.

A Fig. 3.4 retrata de forma simplificada o processo de alocação de margem em um transmissor DSL genérico. O valor final da margem é limitado, entre outros parâmetros pela máxima potência disponível. Valores excessivamente altos de margem na rede geram níveis de *crosstalk* mais altos, e podem ter um efeito negativo tanto na estabilidade de outras linhas, como diminuir os níveis de desempenho (taxa de bits) gerais.

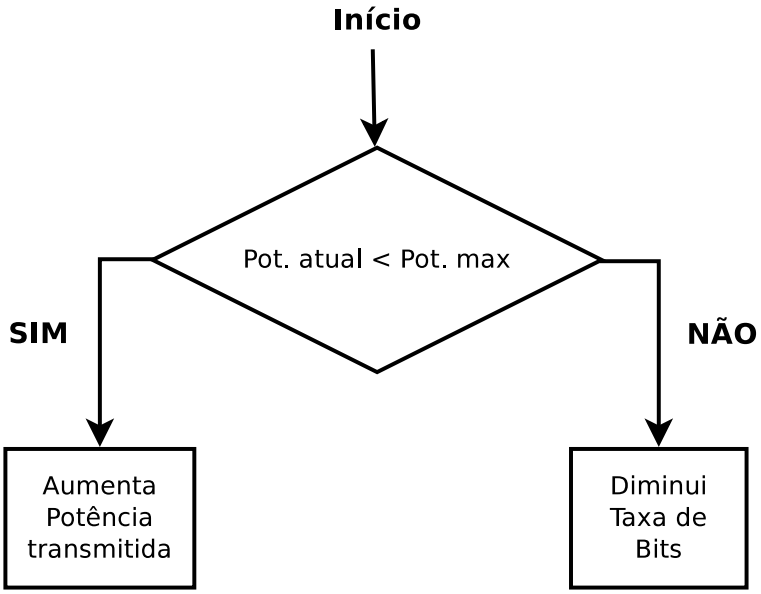


Figura 3.4: Fluxograma representando o processo de alocação de margem num transmissor DSL genérico.

Levando em consideração que problemas de instabilidade repetem-se com frequência nas redes de operadoras DSL pelo mundo, e a deficiência da abordagem simplista descrita anteriormente, pode-se fundamentar a existência da estratégia de otimização escolhida para ilustrar o uso do protótipo desenvolvido neste trabalho. A técnica conhecida como AMA (*Automatic Margin Adaptation*) [12, 33, 39] realiza ajuste automático das margens alvo, o principal parâmetro de controle com influência sobre a estabilidade em sistemas DSL.

Em poucas palavras, a adaptação automática de margem funciona ajustando as configurações de cada linha, de acordo com índices de estabilidade. Linhas consideradas instáveis são reconfiguradas para utilizar valores mais conservadores de margem, vindo então a oferecer mais resistência à degradação da SNR no canal.

Na implementação da estratégia AMA deste trabalho, definiu-se cinco possíveis valores para margem alvo (ver Fig. 3.5). Para cada linha, a estratégia mantém uma contagem do

número de eventos de retreino. Caso esta contagem ultrapasse um limiar (dois retreinos por dia), declara-se instabilidade e o perfil de linha é alterado, para um valor maior de margem. Caso o último estado seja alcançado por uma linha, a estratégia não toma ações posteriores, de forma a não degradar o desempenho geral do sistema com valores muito altos de margens, evitando um efeito cascata, onde o aumento do nível de crosstalk iria disparar retreinos em outras linhas, que por sua vez, teriam que também aumentar os valores de suas margens alvo.

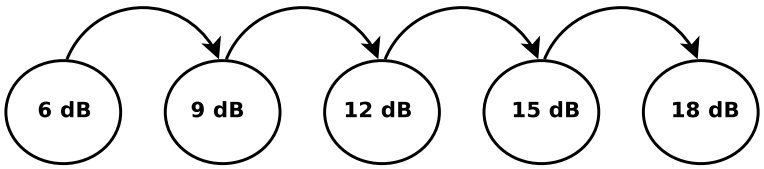


Figura 3.5: Diagrama de estados, mostrando os valores de margem utilizados na implementação do AMA. As transições são disparadas por número excessivo de retreinos durante um dia.

## 3.3 O Setup Experimental

Tendo descrito os blocos constituintes do *software* desenvolvido, a terceira parte deste capítulo trata do cenário de testes montado no laboratório e descreve as medidas adotadas para geração de estímulos para a rede DSL que pudessem, com um determinado grau de fidelidade, reproduzir condições encontradas em uma instalação comercial da tecnologia. Todas as atividades que envolvem equipamentos e procedimentos experimentais descritas nesta seção e no Capítulo 4 foram executadas no Laboratório de Eletromagnetismo Aplicado da Universidade Federal do Pará.

A Fig. 3.6 mostra os equipamentos e *softwares* envolvidos nos experimentos de laboratório. As próximas seções descrevem as dificuldades e soluções adotadas para construir um ambiente de teste capaz de gerar estímulos de tráfego e ruídos, bem como executar o *software* DLM para cerca de vinte linhas.

### 3.3.1 Tráfego IP em redes DSL

Os equipamentos DSL disponíveis em laboratório (DSLAMs Ericsson EDN612 e CPEs de diversas marcas) utilizam ATM como protocolo da camada de enlace, mais especificamente AAL5 (ATM *Adaptation Layer 5*) para transporte de datagramas IP.

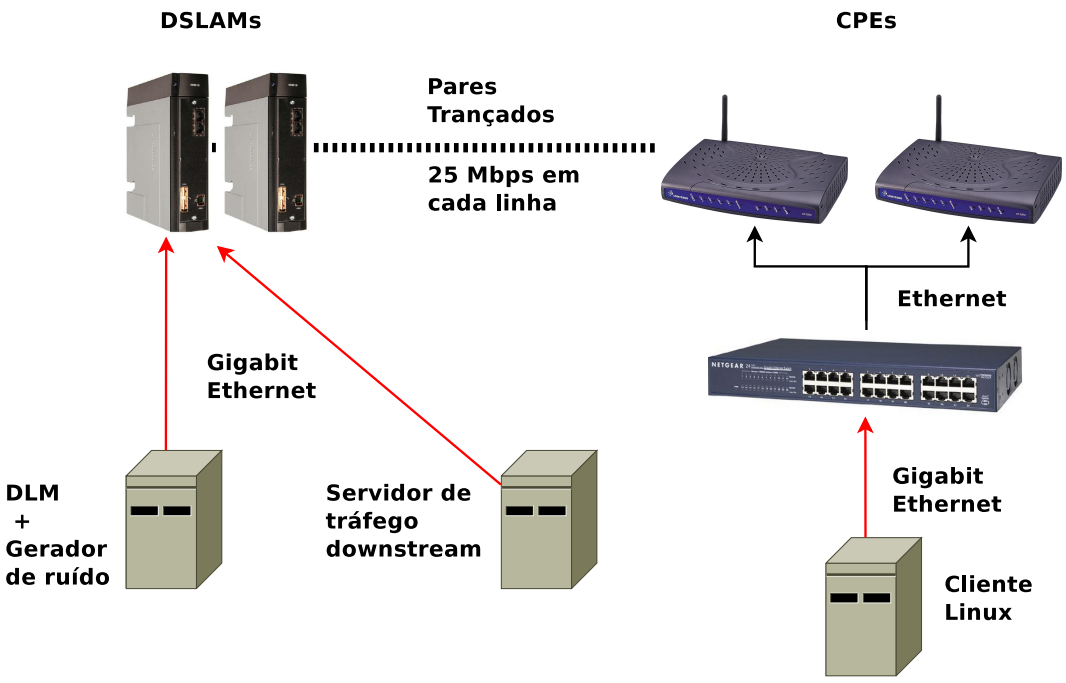


Figura 3.6: Visão geral dos componentes envolvidos na execução dos experimentos em laboratório. São utilizados três computadores além de um switch, modems e DSLAMs.

Entre os terminais de uma conexão DSL estabelece-se (no mínimo) um circuito virtual, identificado por um par de valores chamados VPI (*Virtual Path identifier*) e VCI (*Virtual Circuit Identifier*). Estes valores são utilizados em redes ATM para fins de roteamento.

O protocolo ATM dá suporte ao conceito de classes de tráfego (CBR - *Constant Bit Rate*, VBR - *Variable Bit Rate*, ABR - *Available bit rate* e UBR - *Unspecified bit rate*) [40,41]. Aliado às classes de tráfego, as operadoras podem definir diversos parâmetros de QoS (*Quality of Service*), o que possibilita o fornecimento de diversos serviços sobre um enlace DSL (voz, vídeo e dados).

Seguindo a tendência atual do núcleo das redes de operadoras migrar para tecnologia IP, os DSLAMs disponíveis são capazes de interfacear com o *backbone* de uma operadora utilizando IP diretamente (equipamentos mais antigos "falavam" somente ATM, desde os enlaces DSL até o *backbone*). Desta forma configura-se o quadro mostrado na Fig.3.7, que descreve os protocolos de rede envolvidos num enlace DSL completo.

Na sua interface com o mundo exterior, os DSLAMs utilizam uma única conexão gigabit ethernet com *VLAN tagging*. Em termos gerais, *tags VLAN* são identificadores inseridos em um quadro *ethernet*, através dos quais é possível para várias redes lógicas compartilhar uma única interface física. A utilização de VLANs possibilita que vários tipos de tráfego

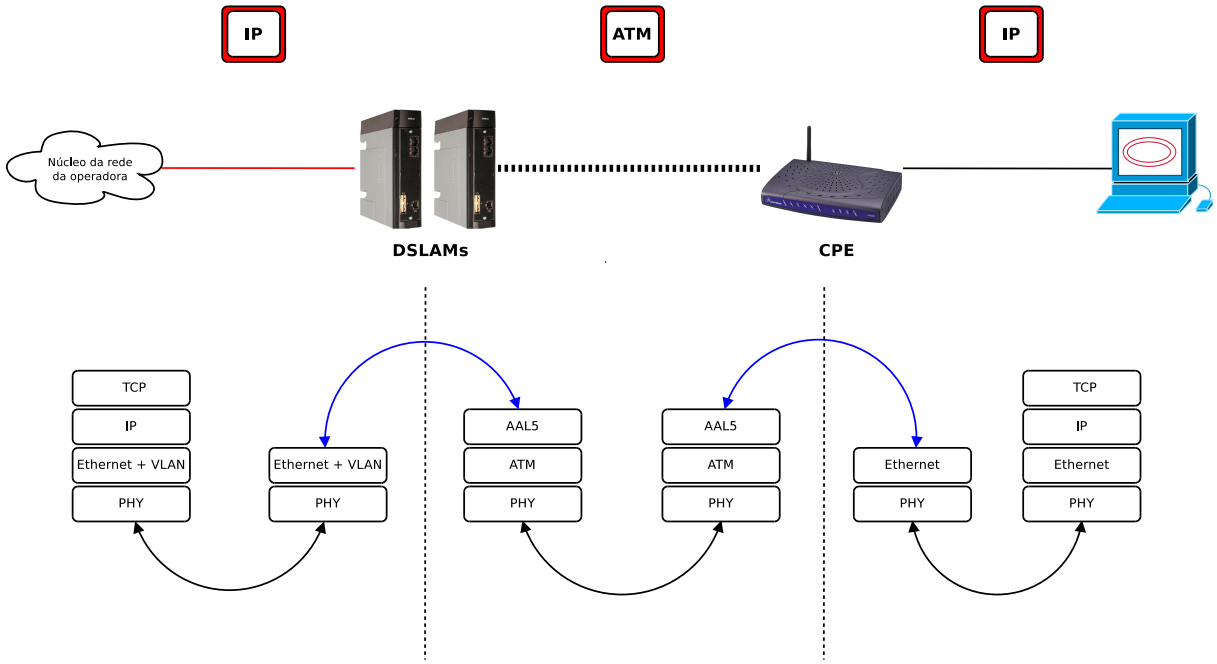


Figura 3.7: Conjunto de protocolos utilizados num enlace DSL completo. A tecnologia ATM é utilizada somente entre o DSLAM e os CPEs, possibilitando à operadora fornecer diferentes serviços, com seus próprios requisitos de QoS. A pilha de protocolos IP está presente no núcleo da rede da operadora e nas máquinas clientes. As setas em azul representam conversões entre datagramas IP e células ATM, realizadas pelos equipamentos DSL.

sejam tratados de acordo com sua natureza e mapeados de forma eficiente nos serviços que uma operadora pode oferecer. Um exemplo da utilização de *tags* para provisão de serviço é representada na Fig. 3.8.

O enlace DSL, com os equipamentos disponíveis para a realização deste trabalho pode ser considerado transparente. Isso significa que, o computador do usuário e o servidor que este deseja alcançar devem estar na mesma subrede (exemplo: 192.168.0.1/24 e 192.168.0.2/24). Em outras palavras, Os endereços IPs dos DSLAMs e CPEs são efetivamente desconsiderados, e estes funcionam meramente como pontes (*bridges*) entre os enlaces ethernet e o enlace DSL[2].

A próxima seção descreve como as características do equipamento DSL disponível foram exploradas para a construção de um *setup* para geração de tráfego compacto, que atendesse os requisitos de projeto do protótipo descrito neste trabalho.

[2]Os endereços IP dos CPEs e DSLAMs, servem somente para funções de configuração e não para trafegar dados no enlace DSL.

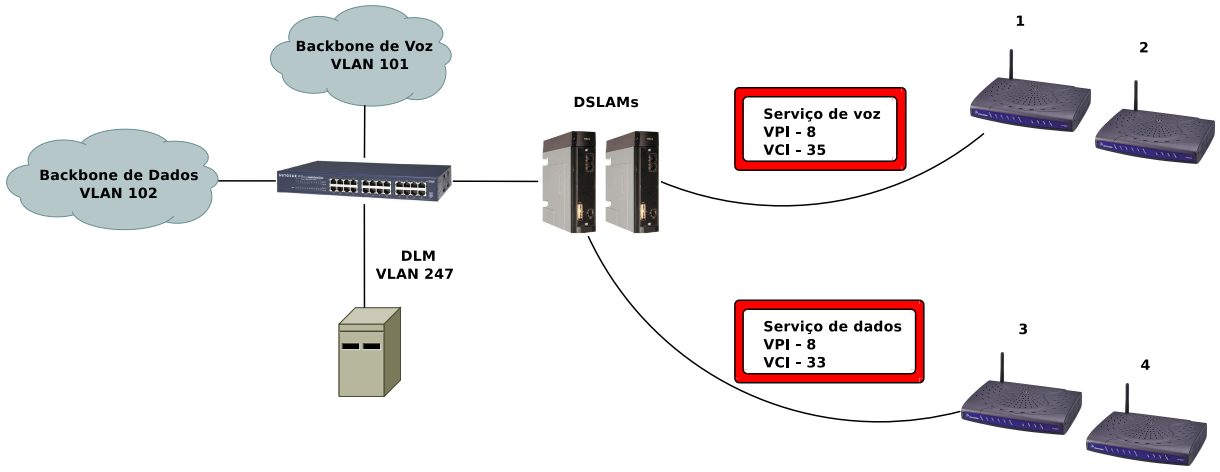


Figura 3.8: Na cenário representado, um DSLAM está ligado através de sua única porta *ethernet* a três redes, utilizando um *switch*. Cada rede é identificada por uma VLAN. Os pacotes provenientes da VLAN 101 são mapeados para o serviço de voz oferecido pela operadora e direcionado somente aos assinantes 1 e 2. Pacotes de dados, marcados com a VLAN 102 somente são enviados aos usuários 3 e 4. Os pacotes da VLAN 247 são especiais, contendo informações de controle, como por exemplo, comandos enviados pelo DLM.

#### 3.3.1.1 Construindo um *setup* compacto para geração de tráfego

Como afirmado anteriormente, um dos grandes diferenciais do protótipo proposto neste trabalho é a aplicação contínua de técnicas de otimização DSM, sem interromper desnecessariamente a conexão do usuário com a internet. Para testar esta característica e simular o comportamento dinâmico do tráfego em redes DSL, projetou-se uma solução de geração de tráfego que pudesse ser implementada no laboratório, com equipamentos comerciais, com o menor custo possível. Os objetivos a serem alcançados pela solução eram os seguintes:

- O número alvo de linhas (usuários DSL) é 23.
- Utilizar somente dois computadores para geração do tráfego em todas as linhas.
- Todas as linhas podem estar ativas (carregando pacotes IP) em pelo menos uma fração do tempo do experimento. Um software deve controlar a geração de tráfego de forma a simular o comportamento dinâmico da rede. A existência de momentos de inatividade é importante pois são aproveitados pelas estratégias para executar ações de otimização.

Para alcançar os objetivos descritos acima, os DSLAMs foram ligados à um único PC ("servidor de tráfego" na Fig. 3.6). Vinte e três (23) modems foram conectados através de

um switch de camada 2 (*gigabit ethernet*, 24 portas), a outro computador ("cliente Linux" na Fig. 3.6). Utilizou-se o sistema operacional Linux em ambos os computadores.

*Switches* de camada 2 mantém uma tabela sobre quais endereços MAC (*Media Access Control*) podem ser alcançados utilizando cada uma de suas portas. Desta forma, os pacotes só são transmitidos às portas diretamente envolvidas na comunicação (exceto pacotes de *broadcast*). Endereços MAC, idealmente identificam de maneira única uma interface de rede física. Considerando o *setup* mostrado na Fig. 3.6, que possui somente duas interfaces de rede físicas, surgem as seguintes questões:

1. A tabela de roteamento do *switch*, que relaciona a porta a ser utilizada para alcançar um determinado MAC de destino é preenchida através do protocolo ARP (*Address Resolution Protocol*. Com um único endereço MAC (o endereço do servidor) ligado à 23 portas do switch através dos CPEs, não se pode determinar com certeza por qual dos enlaces a resposta à requisição ARP retornará primeiro, e consequentemente qual linha DSL será utilizada para carregar o tráfego do cliente para o servidor. Como garantir então, que os pacotes no sentido *upstream* sejam enviados por um enlace DSL específico?

2. O roteamento de pacotes no sentido *downstream* é realizado pelos DSLAMs e é baseado em endereços IP. No DSLAM, endereços IP de destino são mapeados em linhas DSL (exemplo: o endereço 192.168.0.2/24 pode ser encontrado pela linha 2, enquanto que 192.168.0.5/24 pode ser encontrado pela linha 5). Como garantir que somente um enlace específico seja utilizado (evitando duplicação e ou ocupação indevida de outros enlaces)?

A solução para os dois problemas é baseada nos seguintes conceitos: **interfaces virtuais, IP *aliasing*, VLANS e MAC *spoofing***.

- **Interfaces de rede virtuais** são uma característica suportada pelo sistema operacional Linux, que permite a criação, para o *kernel*, de interfaces de rede que não necessariamente estão ligadas a um *hardware*. Interfaces de rede virtuais servem, por exemplo para a atribuição de múltiplos endereços IP a uma mesma interface física. Um outro exemplo de aplicação é a modificação de alguns campos dos pacotes antes de serem enviados por uma interface física. Exemplos de interfaces virtuais incluem também dispositivos que representam conexões ppp (*point-to-point protocol*).

- O termo **IP *aliasing*** refere-se à prática que permite a uma única placa de rede participar de várias subredes ao mesmo tempo (a mesma placa passa a ser identificada por vários endereços IP). Apesar disso, o endereço MAC reportado por interfaces *alias* é igual ao endereço da interface física. Interfaces *alias* são um tipo de interface virtual.

- O suporte à VLANs no *kernel* Linux também é implementado através de interfaces virtuais. Interfaces VLAN e interfaces *alias* possuem as mesmas capacidades e estão sempre ligadas à alguma interface física. A diferença está no fato de que a todos os pacotes transmitidos por uma interface VLAN é adicionada a *tag* (número identificador) correspondente. Além disso, interfaces VLAN podem ser configuradas para informar um endereço MAC diferente do endereço da interface física. Esta técnica é conhecida como **MAC *spoofing***.

O projeto final, que permitiu a geração de tráfego em ambos os sentidos, com várias linhas ativas ao mesmo tempo, e ativando enlaces específicos está retratada na Fig. 3.9.

No servidor de tráfego, que representa o *backbone* da operadora, configurou-se 23 diferentes interfaces VLAN, sendo que a cada uma foi atribuído um endereço MAC artificial (MAC *spoofing*). Observando a Fig. 3.9, a caixa mais a esquerda mostra os endereços IP, *tags* VLANs e valores de MAC escolhidos para cada interface.

No computador cliente, criou-se 23 interfaces *alias* diferentes. Cada interface *alias* foi configurada para utilizar IPs na mesma subrede da VLAN correspondente. Observando a Fig. 3.9, a caixa mais a direita mostra os endereços IP atribuídos à cada interface virtual (eth0:1 até eth0:23).

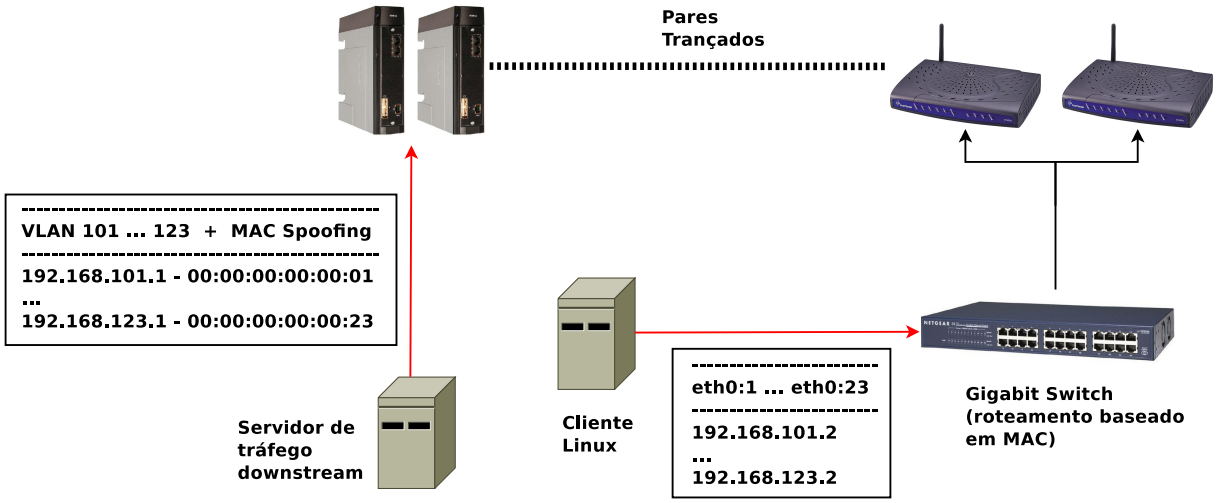


Figura 3.9: Estrutura e configurações das interfaces de rede necessárias ao *setup* de geração de tráfego compacto.

A seguir, são dados dois exemplos que descrevem o caminho dos pacotes através do *setup* desenvolvido, para transmissão *downstream* e *upstream*.

#### 3.3.1.2 Pacotes no sentido *downstream*

1. Supondo que um pacote deve ir do servidor de tráfego para o cliente usando a linha 16. O endereço de origem é 192.168.116.1, e o de destino é 192.168.116.2. Na máquina de origem, é sabido previamente (através de tabelas ARP fixas) que o IP de destino corresponde ao MAC da interface eth0:16 (00:21:5a:2f:fb:d7)[3].

2. A tabela de rotas do servidor de tráfego determina que todos os pacotes com destino na subrede 192.168.116.0 devem ser enviados pela interface VLAN116, recebendo a *tag* correspondente (116) antes do envio.

3. No DSLAM, tráfego identificado pela *tag* 116 é redirecionado para a linha 16 (todas as *tags* são retiradas antes do envio no canal DSL). O pacote viaja pelo enlace DSL e chega ao *switch*. No *switch* é determinado que todos os pacotes com o MAC de destino 00:21:5a:2f:fb:d7 devem sair pela porta 24 (porta a qual o computador cliente está ligado)[4].

4. O pacote é entregue ao à interface eth0:16 do PC cliente.

#### 3.3.1.3 Pacotes no sentido *upstream*

1. Supondo que um pacote deve ir do PC cliente para o servidor de tráfego utilizando a linha 10. O endereço de destino é 192.168.110.1. Através de requisições ARP, determina-se que este endereço IP corresponde ao MAC 00:00:00:00:00:10.

2. A tabela de roteamento do cliente envia o pacote pela interface eth0:10 (efetivamente usando a interface física eth0).

3. No *switch*, determina-se que os pacotes com destino ao MAC 00:00:00:00:00:10 devem ser enviados pela porta 10.

4. A porta 10 do *switch* entrega os pacotes ao CPE ligado à linha 10 do DSLAM. O pacote chega à interface *ethernet* do modem e é prontamente transmitido pelo enlace DSL.

5. No DSLAM, o tráfego vindo da linha 10 recebe a *tag* 110 e é enviado pela única porta *gigabit ethernet* do mesmo, chegando à interface VLAN110 do servidor de tráfego.

[3]É importante lembrar que o MAC de todas as interfaces *alias* do cliente são iguais ao MAC da interface física. Em outras palavras, o MAC 00:21:5a:2f:fb:d7 de fato corresponde ao MAC da interface eth0.

[4]Considerando o raciocínio exposto em nota anterior, como o MAC é o mesmo para todas as interfaces *alias* do cliente, efetivamente todo o tráfego que chega ao switch é redirecionado para a porta a qual o cliente está conectado (24).

#### 3.3.1.4 O *software* gerador de tráfego

O *setup* para geração de tráfego possui também um componente de *software*, que realiza o agendamento dos períodos de atividade / inatividade no enlace. A geração dos pacotes é efetivamente realizada pelo software **iperf** [42]. Um *script* desenvolvido em linguagem Python gerencia diversas *threads* que baseadas num gerador de números pseudo-aleatórios com distribuição uniforme, intercala períodos de atividade (onde o iperf é executado) e inatividade em cada linha DSL. A Fig. 3.10 traz uma representação do algoritmo do *software* gerador de tráfego (Uma instância do algoritmo é executada para cada linha de maneira concorrente, o que possibilita a existência de vários enlaces ativos ao mesmo tempo).

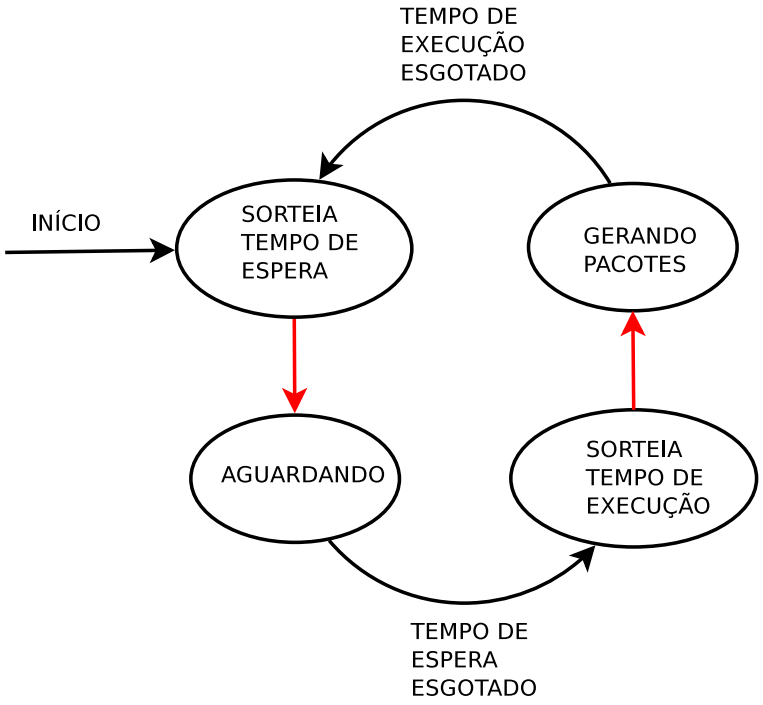


Figura 3.10: Algoritmo para geração de tráfego. As transições não associadas a uma condição (em vermelho), ocorrem instantaneamente, assim que os processos de origem terminam de ser executados.

Encerra-se assim a descrição do *setup* criado para geração de tráfego. O projeto aqui descrito é utilizado na geração de parte dos resultados experimentais apresentados no Capítulo 4. A seguir, é discutido a outra parte importante do *setup* experimental, responsável pela geração de ruído.

### 3.3.2 Geração de ruído

Assumindo que as condições de ruído de fundo e ruídos provenientes de fontes externas ao ambiente controlado de um laboratório não mudem muito durante o período de execução dos experimentos, a existência de um gerador de ruído controlado por *software* possibilita a execução de testes em que as condições de ruído são reprodutíveis.

O protótipo para otimização de sistemas DSL descrito neste trabalho beneficia-se da existência de tal mecanismo controlado para geração de ruído. Escolheu-se implementar um gerador que pudesse, através de *software* determinar os momentos exatos em que os sinais interferentes seriam injetados nos enlaces, criando a possibilidade de comparação justa de diferentes estratégias de otimização.

O gerador de ruído utilizado nos experimentos com o DLM utiliza os próprios CPEs para gerar *crosstalk* que venha a desestabilizar as linhas gerenciadas. Do total de linhas disponíveis, separam-se algumas que serão utilizadas somente para geração de ruído. O software desenvolvido utiliza então o bloco SNMP(ver Seção 3.2.3) e envia comandos para os DSLAMs ativarem e desativarem os sinais DSL nas linhas.

O algoritmo para o software que controla a geração de ruído está representado na Fig. 3.11. Como no caso do software gerador de tráfego, uma instância deste algoritmo é executada para cada linha interferente, possibilitando que vários interferentes estejam ativos ao mesmo tempo. A semente do gerador de números pseudo-aleatórios pode ser especificada pelo usuário, criando assim a possibilidade de reproduzir de forma razoável as condições de ruído em diferentes experimentos. O gerador de números pseudo-aleatórios utiliza uma distribuição de probabilidades uniforme.

Este capítulo descreveu o projeto e a implementação de um protótipo para otimização contínua em sistemas DSL. Foram apresentados aqui os componentes deste protótipo, tanto de *software* quanto *hardware*. O próximo capítulo trata da utilização do protótipo desenvolvido em duas aplicações práticas relacionadas à otimização em sistemas DSL, e apresenta resultados de experimentos realizados com o *setup* experimental aqui descrito.

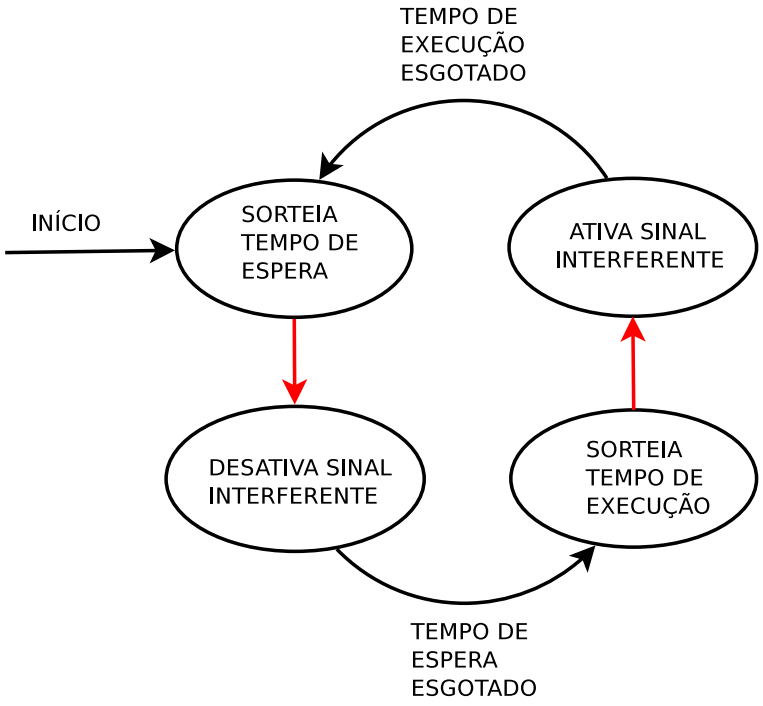


Figura 3.11: Algoritmo básico para a geração de ruído em uma determinada linha. As transições não associadas a uma condição (em vermelho), ocorrem instantaneamente, assim que os processos de origem terminam de ser executados.

# Capítulo 4

# Resultados Experimentais

## 4.1 Introdução

Este capítulo apresenta resultados experimentais obtidos utilizando o protótipo para otimização de sistemas DSL descrito no Capítulo 3. Duas contribuições distintas são apresentadas:

- Na Seção 4.2, descreve-se a utilização da estratégia de otimização AMA, com o intuito de estabilizar um sistema DSL com 10 usuários.
- Na Seção 4.3, descreve-se a aplicação de partes do *software* DLM para comparação do desempenho de algoritmos DSM nível 2, baseados em informação de canais de *crosstalk* estimados e medidos.

## 4.2 Adaptação automática de Margem em um sistema DSL prático

Os objetivos do primeiro estudo de caso, que trata do uso do protótipo na otimização de uma rede DSL baseado em adaptação de margem são os seguintes:

- Estabilizar uma rede DSL com 10 usuários, através da execução da estratégia de otimização AMA, implementada como descrito na Seção 3.2.7.1.
- Avaliar os efeitos colaterais da aplicação da estratégia sobre o desempenho do sistema.

- Pôr em teste todos os subsistemas do protótipo (DLM, gerador de tráfego e ruídos).

### 4.2.1 Topologia da rede DSL

A topologia escolhida para a realização do teste de aplicação da estratégia AMA (ver Fig. 4.1) baseia-se num caso de topologia próximo-distante (*near-far*) comumente utilizada em trabalhos que tratam de algoritmos DSM e DSL por evidenciar os efeitos do FEXT (*far-end crosstalk*) [5–7, 10].

As linhas longas, ligadas diretamente a uma central da operadora (comumente chamada CO - *central office* em trabalhos relacionados) compartilham uma seção de cabo com linhas mais curtas, provenientes de um terminal remoto (RT, *remote terminal*). Ao chegar aos receptores (linhas 1 a 10 na Fig. 4.1), os sinais provenientes do CO já estão bastante atenuados. Devido ao menor comprimento do enlace que sai RT, os sinais enviados por estes pares sofrem pequena atenuação. Quando os sinais interferentes atingem os receptores (linhas de 1 a 10), a potência do *crosstalk* é alta, o que limita o desempenho das linhas mais longas e pode potencialmente causar instabilidade.

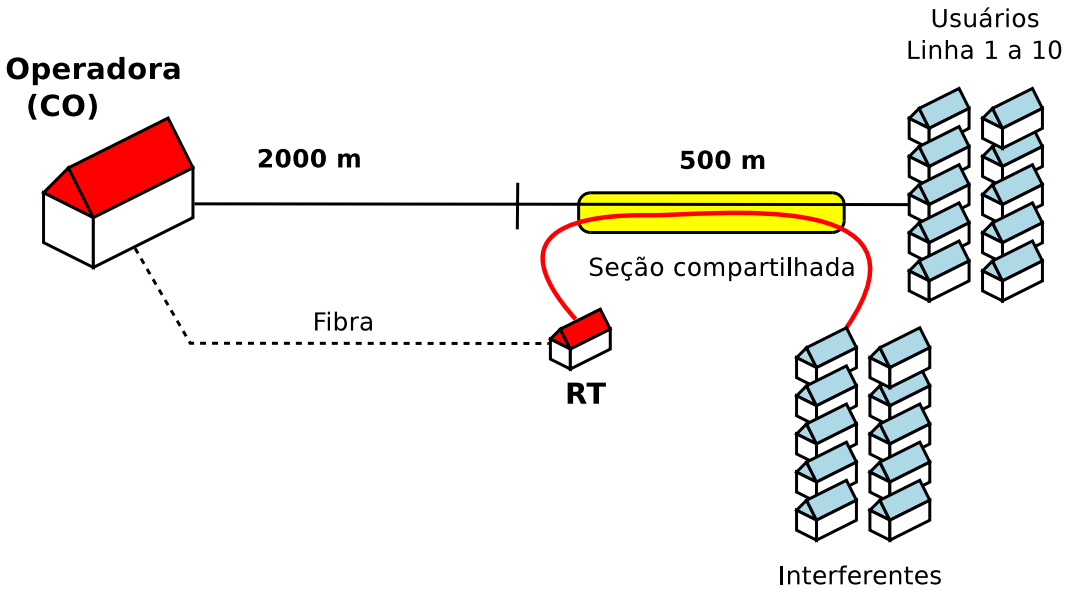


Figura 4.1: A topologia escolhida para a realização do teste de aplicação da estratégia AMA. Dez usuários (ligados ao CO) e dez interferentes (ligados ao RT) são utilizados no experimento. As linhas longas são o alvo da otimização. As linhas mais curtas são utilizadas pelo gerador de ruído.

A seguir é discutida a metodologia utilizada para a execução dos experimentos.

### 4.2.2 Metodologia

Para os fins do experimento, os usuários de interesse são aqueles que estão conectados às linhas provenientes do CO, numerados de 1 a 10 na Fig. 4.1. As outras dez linhas, ligadas ao RT são utilizadas pelo gerador de ruído para causar instabilidade na rede.

Adotou-se dois retreinos por dia como limiar para detecção de instabilidade. Ao detectar instabilidade a estratégia AMA aumenta o valor da margem alvo na linha, processo representado na Fig. 3.5). O limiar de detecção foi escolhido baseando-se em requisitos de estabilidade sugeridos em [39], para provisão de serviços de dados (internet) e vídeo-sobre-IP. Os contadores de retreino são zerados a cada dia, o que implica que ao fim de um dia, uma linha se manterá no mesmo estado AMA (com o mesmo valor de margem alvo) caso a contagem seja menor que o limiar de detecção.

Considera-se uma linha **estável** caso esta não apresente retreinos por um período de dois dias consecutivos. Caso uma linha atinja o último estado AMA e continue apresentando retreinos, esta é declarada instável (a estratégia "desiste" de tomar ações nesta linha).

Utilizou-se o gerador de tráfego descrito no último capítulo em todas as linhas de interesse, bem como o gerador de ruídos nas linhas interferentes. A geração de ruído foi realizada em intervalos escolhidos por um gerador de números pseudo-aleatórios com distribuição de probabilidades uniforme, que escolhia intervalos de espera e execução (ver Fig. 3.11) entre 3 e 8 minutos. Cada um dos dez interferentes podia ser ativado ou desativado independentemente.

Configurou-se o bloco DSL Logger para registrar informações sobre o estado das linhas a cada 20 segundos. As linhas foram monitoradas por aproximadamente cinco dias (de 19/03 às 17:47 até 24/03 às 10:53), totalizando 20346 observações para cada linha monitorada.

### 4.2.3 Resultados

Considerando o primeiro objetivo do experimento, **estabilizar a rede**, a partir dos dados listados na Tabela 4.1, pode-se afirmar que a estratégia implementada alcançou sucesso. Todas as linhas de interesse atingiram o critério de estabilidade adotado. Nenhuma das linhas alcançou o último estado da estratégia (18 dB).

Tendo atendido aos critérios de estabilidade é importante avaliar as perdas, nas taxas de bits, causadas pela utilização da estratégia de ajuste de margem. A Fig. 4.2 ilustra os valores de taxa de bits para todas as linhas de interesse. São apresentados valores de taxa colhidos nas primeiras observações do experimento, já com a presença de um interferente e no

Tabela 4.1: Valores finais de margem alvo para cada linha, e data do último retreino. Todas as linhas alcançaram o critério de estabilidade. As linhas 4 e 7 apresentaram um número um pouco mais acentuado de retreinos, refletido aqui no fato de terminarem com margens alvo mais altas.

| Início: 2010-03-19 17:47:32 | | |
|---|---|---|
| Fim: 2010-03-24 10:53:29 | | |
| Linha | Último retreino | Margem alvo final (dB) |
| 1 | 2010-03-21 19:17:14 | 12 |
| 2 | 2010-03-20 19:16:53 | 12 |
| 3 | 2010-03-21 19:17:14 | 12 |
| 4 | 2010-03-21 19:17:14 | 15 |
| 5 | 2010-03-20 19:16:53 | 12 |
| 6 | 2010-03-20 19:16:53 | 12 |
| 7 | 2010-03-21 19:17:14 | 15 |
| 8 | 2010-03-21 19:17:14 | 12 |
| 9 | 2010-03-20 19:16:53 | 12 |
| 10 | 2010-03-21 19:17:14 | 12 |

final, representados respectivamente em verde, azul e vermelho.

Os valores de taxa de bits iniciais (em verde na Fig. 4.2) foram recolhidos após 3 minutos do início do experimento. Devem ser considerados como as máximas taxas de bit alcançáveis no cenário utilizado. No momento em que os valores das taxas de bit iniciais foram colhidos, nenhum modem interferente (ligado ao RT) estava ativo. As linhas de interesse estavam limitadas somente pelo *crosstalk* entre elas mesmas e ruídos externos (ruído de fundo, rádio frequência, entre outros).

As barras em azul na Fig. 4.2 representam as taxas de bit aos 6 minutos de execução do experimento, quando um interferente já encontra-se ativo. Esses valores são apresentados aqui para tornar a comparação entre taxas iniciais e finais mais justa. Apesar da linha 1 manter-se com a mesma taxa da inicialização, todas as outras tiveram que retreinar devido ao intenso *crosstalk* gerado pelo interferente, retomando a transmissão em uma taxa mais baixa.

Os valores de taxa de bits finais estão representados em vermelho na Fig. 4.2. Naturalmente são mais baixos que os valores iniciais e com um interferente. É importante

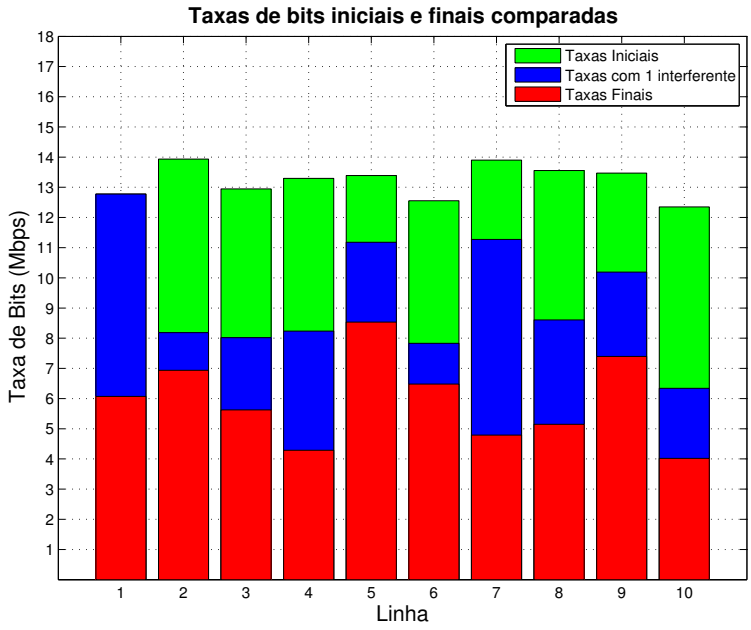


Figura 4.2: Comparação das taxas de bits alcançadas pelas linhas de interesse. As barras em verde representam as taxas iniciais, quando nenhum interferente estava ativo. As taxas em azul foram reportadas pelas linhas de interesse aos 6 minutos de execução do experimento, com um interferente ativo. Em vermelho, encontram-se as taxas no fim do experimento, com todas as linhas estáveis. Na linha 1, as taxas iniciais e com um interferente ativo foram as mesmas, por esse motivo a barra verde aparece encoberta pela barra azul.

notar que, com os valores de taxa finais as linhas estão estáveis. Isto significa que mudanças no número de interferentes ativos não causará mais retreinos, pois os valores de margem foram ajustados pela estratégia de otimização.

A comparação entre os valores de taxa de bits finais e com a presença de um interferente, apresenta um decréscimo aceitável na maioria das linhas, a exceção sendo as linhas 4 e 7, que foram aquelas que tiveram mais dificuldade para se estabilizar, segundo dados da Tabela 4.1. As diferenças do impacto causado pela presença de interferentes sobre o desempenho de uma determinada linha pode ser atribuída em sua maior parte pela variação nas funções de transferência de *crosstalk* ente os diferentes pares num *binder*. Isto explica por que, por exemplo, as linhas 4 e 7, apesar de terem as mesmas características físicas de outras linhas sofreram mais retreinos durante o experimento.

Os valores de taxa de bits para o caso com um interferente e para o caso estável (no fim do experimento), estão listados na Tabela 4.2 para facilitar a comparação. As linhas 4, 7 e

Tabela 4.2: Comparação dos valores de taxas de bit alcançados pelas linhas de interesse em dois casos: com a presença de um interferente ativo e no fim do experimento, com todas as linhas estáveis. As linhas 4 e 7 apresentaram o pior desempenho.

| Linha | Taxa com um interferente (Mbps) | Taxa final linhas estáveis (Mbps) | Variação (%) |
|---|---|---|---|
| 1 | 12.77 | 6.07 | -52 |
| 2 | 8.18 | 6.93 | -15 |
| 3 | 8.01 | 5.62 | -29 |
| 4 | 8.23 | 4.28 | -47 |
| 5 | 11.17 | 8.53 | -23 |
| 6 | 7.82 | 6.47 | -17 |
| 7 | 11.27 | 4.79 | -57 |
| 8 | 8.60 | 5.14 | -40 |
| 9 | 10.19 | 7.39 | -27 |
| 10 | 6.33 | 4.01 | -36 |

1 estão destacadas em vermelho por serem os piores casos. Entretanto, deve-se considerar que a taxa de bits da linha 1 na segunda coluna está alta temporariamente, já que esta resistiu à adição do primeiro interferente, mas viria a sofrer retreinos com a adição dos próximos. É também importante ressaltar que caso o interferente acionado fosse outro, a taxa na linha 1 poderia estar no nível das demais (devido à variabilidade nas funções de transferência de *crosstalk*).

O caso ideal de funcionamento da estratégia AMA é aquele em que o valor médio da margem (SNR *margin*) fica **abaixo** do valor da margem alvo especificado. Caso contrário, podem estar ocorrendo duas situações:

- A potência do ruído diminuiu.
- Parte da potência que está sendo utilizada para manter uma margem excessiva poderia estar sendo utilizada para aumentar a taxa de bits.

A a Fig. 4.3 traz gráficos relacionados aos valores de margem e taxa de bits recolhidos durante todo o experimento, na linha 1. Observando a Fig. 4.3, da amostra 10000 em diante, pode-se notar que o valor **médio** da margem está acima do valor da margem alvo, o

que significa que na maior parte do tempo, se mantém um valor de margem desnecessário, limitando assim o desempenho do enlace, que fica "preso" em 6 Mbps.

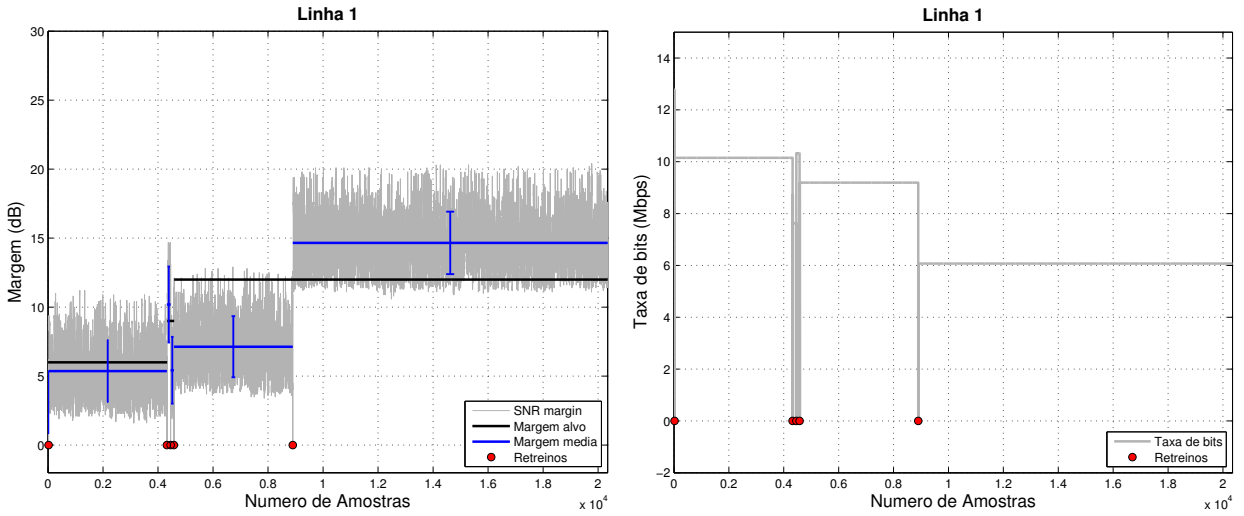


Figura 4.3: Valores de margem e taxa de bits registrados durante todo o experimento na linha 1. Os momentos exatos em que ocorreram retreinos estão marcados em vermelho. A linha em azul representa o valor médio da margem em cada intervalo entre eventos de retreino. As barras verticais em azul equivalem ao desvio padrão da margem nos intervalo entre retreinos consecutivos. A linha em preto representa o valor da margem alvo especificado pela estratégia.

O comportamento visto no gráfico da Fig. 4.3 tem as características do problema chamado *stuck-at low rate*, descrito em [12]. Este problema acontece quando uma linha que já tem um alto valor de margem alvo sofre um retreino eventual (por exemplo, causado por ruído impulsivo, tipo de interferência contra a qual a margem não tem muita eficácia). Ao retreinar, o modem calcula o bitloading considerando uma margem alvo muito alta, sendo muito conservador. Como a rajada de ruído impulsivo era uma condição temporária, a taxa de bits do transmissor fica limitada, até a linha ser reiniciada ou ocorrer outro retreino.

A Fig. 4.4, por outro lado mostra um caso em que as margens alvo determinadas pela estratégia AMA foram suficientes para evitar retreino, sem que ocorresse prejuízos demasiados em termos de taxa de bits. Por exemplo, por volta da amostra 12000, é possível notar que o nível da SNR *margin* se aproxima bastante de 0, o que teria causado um retreino, caso a margem alvo não tivesse sido ajustada pela estratégia para o valor de 12 dB. Com o intuito de fornecer uma visão mais completa sobre o funcionamento da estratégia de otimização em questão, o Apêndice A traz as figuras geradas com os dados colhidos para as demais linhas de interesse.

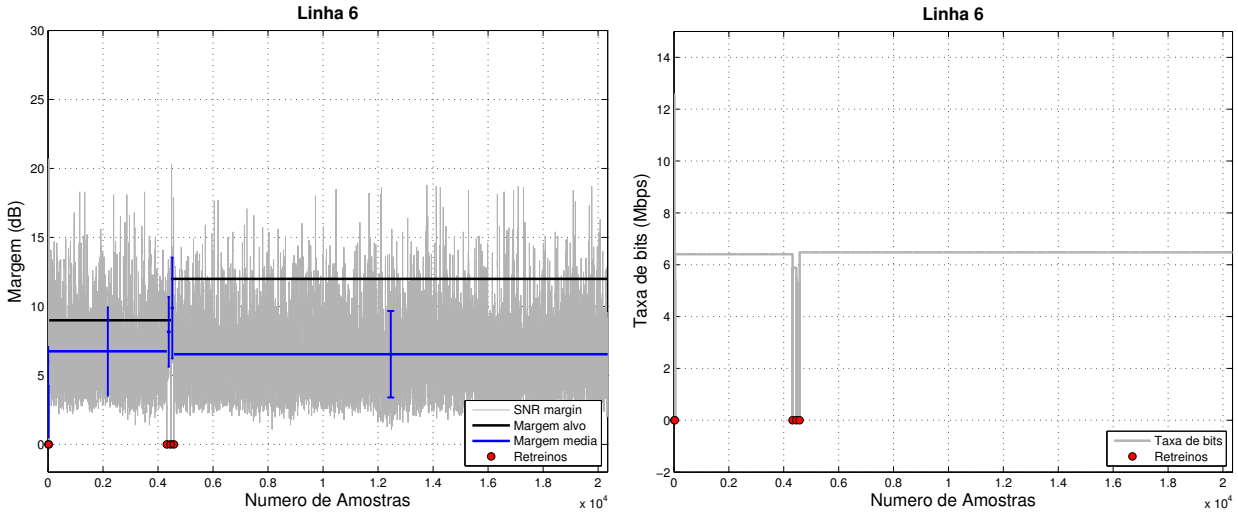


Figura 4.4: Valores de margem e taxa de bits registrados durante todo o experimento na linha 6. Os momentos exatos em que ocorreram retreinos estão marcados em vermelho. A linha em azul representa o valor médio da margem em cada intervalo entre eventos de retreino. As barras verticais em azul equivalem ao desvio padrão da margem nos intervalo entre retreinos consecutivos. A linha em preto representa o valor da margem alvo especificado pela estratégia.

Sobre o funcionamento do protótipo é possível afirmar que todos os componentes interagiram como esperado, servindo como uma base sólida para o funcionamento da estratégia de otimização implementada. Os *setups* para geração de tráfego e geração de ruídos também funcionaram a contento, gerando estímulos durante o período de execução dos testes.

A implementação da estratégia AMA aqui apresentada cumpriu o objetivo de estabilizar as linhas de interesse. Apesar disso, analisando os dados coletados e as figuras no Apêndice A, é possível afirmar que algumas linhas poderiam ter um melhor desempenho quanto às taxas de bits alcançadas.

A estratégia de ajuste automático de margens utilizada neste trabalho para testar o protótipo desenvolvido possui deficiências, sendo que pelo menos uma delas se manifestou nos resultados do experimento (*stuck-at low rates*).

É possível que alguns dos eventos de retreino tenham sido causados por ruído impulsivo, tendo em vista que em alguns momentos, mesmo com margens muito altas, linhas retreinaram. A estratégia AMA, como implementada e discutida aqui é ineficaz no combate aos retreinos causados por ruído impulsivo, pois só altera valores de margem. Para lidar melhor com ocorrências de ruído impulsivo seria preciso também alterar parâmetros como o *interleaver delay* e o INP (*Impulse Noise Protection*).

Um modelo proposto para tratar alguns dos aspectos problemáticos da estratégia AMA pode ser encontrado na Fig. 4.5. Para combater o problema de linhas com taxas de bits "presas" devido ao alto valor de margem alvo, propõe-se que, ao invés de progredir em um único sentido, os estados da estratégia AMA possam ter um caminho de volta (representado em transições vermelhas na Fig. 4.5), efetivamente reduzindo as margens alvo quando possível. Essas transições em sentido contrário seriam disparadas quando detectado que o valor médio da margem está acima do valor de margem alvo durante um determinado período de observação.

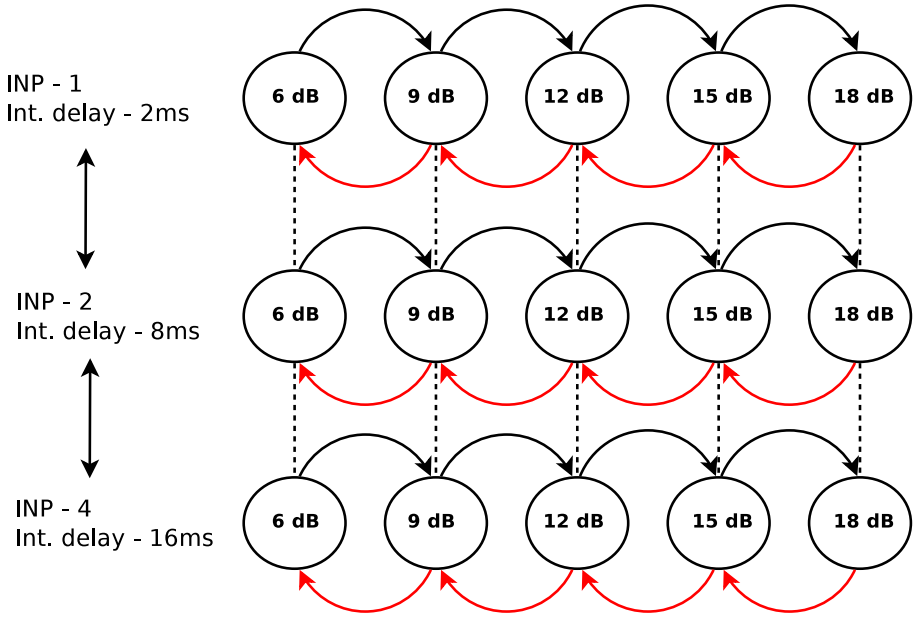


Figura 4.5: Modelo proposto para tratar as deficiências da estratégia AMA implementada neste trabalho. O modelo tem como características transições (setas em vermelho) que diminuem o valor da margem alvo e a inclusão de parâmetros que tratam ruído impulsivo (INP e Int. *delay*).

Um outro aspecto da estratégia proposta é considerar o combate ao ruído impulsivo, alterando parâmetros de INP e *interleaver delay*. Essa mudança acrescentaria mais uma direção no caminho de progressão dos perfis de linha gerados pela estratégia AMA. Ao invés de percorrer a grade representada na Fig. 4.5 somente da direita para a esquerda, a estratégia poderia também descer e subir, garantindo estabilidade e proteção contra ruído impulsivo.

O modelo sugerido pode ser implementado utilizando as facilidades fornecidas pelo protótipo. Como este pode detectar quando as linhas não estão transmitindo tráfego útil, seria possível reiniciá-las, sempre que necessário para combater o problema de *stuck-at low rates*. Para implementar as transições relativas ao ruído impulsivo, utilizaria-se uma segunda variável de controle. Ao invés de contar somente retreinos, contaria-se também o número de

violações de código, que fornecem informações mais úteis para a detecção de ruído impulsivo no *binder*.

A próxima seção trata da aplicação de partes do software DLM para a avaliação do desempenho de um algoritmo DSM nível 2 baseado em informação de canal provida por um estimador.

## 4.3 Desempenho prático de um algoritmo DSM nível 2 baseado em informação de canal estimada

O segundo estudo de caso utilizando o protótipo desenvolvido neste trabalho é apresentado a seguir. O objetivo principal desta seção é fornecer uma comparação do desempenho de um algoritmo DSM nível 2 ISB [7, 23], tendo como entrada informações de canal medidas e fornecidas por um estimador [24]. O estimador, bem como o procedimento experimental descritos aqui fazem uso de alguns blocos do *software* DLM. Uma descrição mais detalhada dos resultados do uso de informação de canal estimada em algoritmos DSM testados em hardware pode ser encontrada em [43].

### 4.3.1 Algoritmos DSM nível 2 e informação de *crosstalk* estimada

A maioria dos algoritmos DSM nível 2 propostos [6–10] assumem que a informação a respeito dos canais de *crosstalk* já está previamente disponível. Em um laboratório, onde ambas as extremidades de um cabo estão acessíveis, as funções de transferência de canais de *crosstalk* pode ser medidas utilizando, por exemplo, um analisador de rede[1]. Entretanto, numa rede DSL real, as extremidades de um cabo podem estar a quilômetros de distância uma da outra. Desta forma, é impraticável e custoso para uma operadora despachar pessoal para executar as medidas em todos os pares de sua rede.

Um estimador prático para canais de *crosstalk*, compatível com os padrões, foi apresentado em [24]. O impacto desse estimador no desempenho de técnicas DSM foi avaliado em [44], baseando-se somente em simulações.

Neste estudo de caso, a avaliação de desempenho de técnicas DSM proposta em [44] é levada um passo adiante. Utilizando um protótipo para aplicação de DSM, avalia-se o desempenho de técnicas DSM nível 2 em equipamentos comerciais, em um cabo real,

[1]Equipamento utilizado para medir a resposta em frequência de um determinado canal.

comparando soluções baseadas em informação de canais de *crosstalk* medidas e fornecidas pelo estimador supracitado.

### 4.3.2 Estimação dos Canais de *Crosstalk*

O método de estimação de canais de *crosstalk* utilizado nos testes de laboratório é brevemente descrito nesta seção. Informações mais detalhadas sobre o método podem ser encontrados em [24]. Assume-se aqui o modelo de sistema descrito na Seção 2.3.

O estimador baseia-se em medidas sequenciais de densidade espectral de potência (PSD) nos receptores (considerando sentido *downstream*), utilizando um sinal interferente gerado em por uma linha de cada vez. Por utilizar um procedimento de duas portas padronizado, referenciado nos padrões como *Loop Diagnostic* [28], o procedimento de estimação pode ser executado e coordenado por um sistema de gerenciamento centralizado.

Fazendo uso de uma notação matricial, pode-se formalizar o método de estimação sequencial como um sistema SIMO (*single-input*, *multiple-output*). Assumindo que somente o $m$-ésimo transmissor está ativo por vez, pode-se expressar a $m$-ésima sequência conforme a Equação 4.1.

$$\mathbf{y}(m) = \mathbf{x}(m)\mathbf{H}(m) + \mathbf{z}(m). \tag{4.1}$$

Aqui $\mathbf{y}(m) = [\bar{y}_1(m)\, \bar{y}_2(m) \ldots \bar{y}_N(m)]$ é a matriz $K \times N$ que contém os sinais recebidos em todos os $K$ subcanais para todos os $N$ receptores, e $\mathbf{H}(m) = [\bar{h}_{1,m}\, \bar{h}_{2,m} \ldots \bar{h}_{N,m}]$ denota a matriz SIMO $K \times N$. A matriz com os $K \times K$ sinais conhecidos do transmissor $m$ tem a forma

$$\mathbf{x}(m) = \begin{pmatrix} x_m^1 & 0 & 0 & 0 \\ 0 & x_m^2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & x_m^K \end{pmatrix}.$$

Em (4.1) o ruído (complexo) adicionado é definido pela matriz $K \times N$ $\mathbf{z}(m) = [\bar{z}_1(m)\, \bar{z}_2(m) \ldots \bar{z}_N(m)]$.

A estimativa da matriz de atenuação FEXT para a sequência $m = 1, 2, ..., N$ pode ser formulada como, [24],

$$\widetilde{|\mathbf{H}|}^{\mathbf{2}}(m) \;=\; \mathbf{P}_x(m)^{-1}\Big(\mathbf{P}_y(m) - \mathbf{P}_z(m_0)\Big), \tag{4.2}$$

Onde $\mathbf{P}_y(m)$, $\mathbf{P}_x(m)$, e $\mathbf{P}_z(m)$ são as matrizes de PSD correspondentes obtidas tomando-se o quadrado do valor absoluto dos elementos de $\mathbf{y}(m)$, $\mathbf{x}(m)$, e $\mathbf{z}(m)$, respectivamente. Em (4.2), $\mathbf{P}_z(m_0)$ denota o ruído de fundo medido sem nenhum transmissor ativo, antes do início da sequência $m$. Partindo de (4.1)–(4.2) segue-se que a estimativa $\widetilde{|\mathbf{H}|}^{\mathbf{2}}(m)$ se torna imparcial (*unbiased*) se $\mathbf{P}_z(m) \approx \mathbf{P}_z(m_0)$. Essa assunção de estacionariedade (temporária) é justificada em pelo menos dois pontos: no caso SIMO nenhum outro interferente está presente, e o canal do par trançado é invariante no tempo. Uma descrição mais detalhada do estimador e seu desempenho pode ser encontrada em [24]. Deve-se enfatizar que uma vez que os canais FEXT não mudam significativamente com o tempo, o processo de estimação intrusivo não precisa ser executado com frequência.

### 4.3.3 Aplicação de DSM nível 2 utilizando o *software* DLM

A maioria dos algoritmos DSM na literatura não lida com os detalhes de implementação de camada física, ou limitações práticas impostas por *hardware* pré-existente. Por exemplo, modems DSM comerciais não podem mudar arbitrariamente a PSD utilizada enquanto estão transmitindo (em outras palavras, estado de *showtime*).

Embora as PSDs não possam ser alteradas no estado de *showtime*, existe um conjunto de parâmetros padronizados, chamados *transmitter spectrum shaping* ($tss_i$), que permitem modificar o formato das PSDs nos serviços ADSL2 e ADSL2+ [28, 29]. Utilizando estes parâmetros, é possível definir uma máscara limite para a PSD, que os modems são obrigados a respeitar durante a transmissão. Infelizmente, as máscaras de PSD limite só podem ser alteradas enquanto a linha de interesse está desabilitada (estado C-IDLE [28]).

O suporte à otimização de sistemas DSL com algoritmos de nível 2 no *software* DLM é implementado através do simulador jDSLsim [22], que é responsável por calcular as máscaras de PSD limite. Os resultados deste bloco, são traduzidos em parâmetros $tss_i$ através de uma técnica descrita em [45].

A Fig. 4.6 representa os blocos envolvidos num processo de otimização DSM nível 2 genérico, baseado em informação de canal estimada. Uma descrição da forma de execução deste tipo de otimização é dada a seguir:

- Inicialmente, as funções de transferência de canais diretos e de *crosstalk* são estimadas utilizando o método descrito na seção anterior. Alternativamente as funções de transferência podem ser medidas utilizando um analisador de rede.
- Um algoritmo DSM é utilizado para calcular um conjunto de PSDs otimizadas (bloco

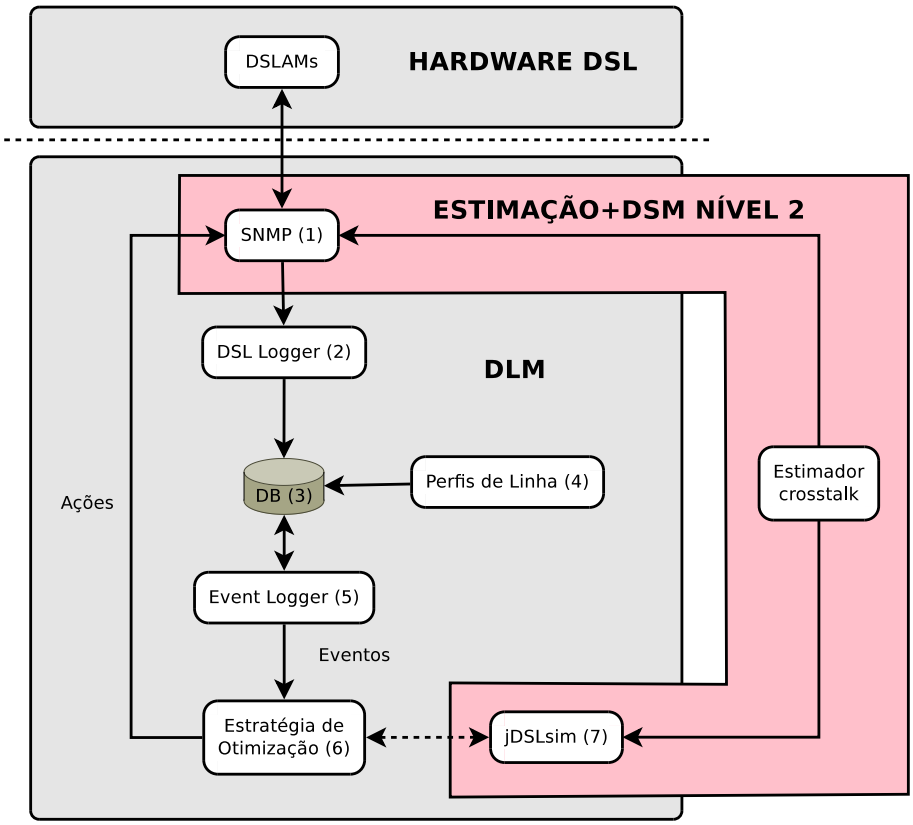


Figura 4.6: Esquema mostrando os blocos do DLM envolvidos na obtenção dos resultados de DSM nível 2 com informações de canais de *crosstalk* estimadas.

jDSLsim) baseado nas quantidades obtidas no primeiro passo.

- As PSDs resultantes do processo de otimização são modificadas [45] para gerar um conjunto de parâmetros de controle válidos (valores de $tss_i$) para serem aplicados nos DSLAMs.

- Finalmente, após a aplicação das máscaras de PSD (realizada utilizando as funções do bloco SNMP), o *protótipo* faz requisições ao DSLAM para avaliar a resposta da rede (taxas de bits, bit load, potência de saída, entre outros).

Existem duas fontes principais de erro no processo. Primeiro, as informações de canal estimadas são afetadas pelo método de estimação e pelo *hardware* utilizado, por exemplo, a implementação do protocolo *Loop Diagnostic* pode variar dependendo do fabricante. Segundo, pode ocorrer erro no mapeamento entre as PSDs calculadas e aquelas reproduzidas pelos modems, já que as PSDs só são obrigadas a não excederem a máscara limite (os níveis de potência de transmissão podem variar abaixo da máscara).

A próxima seção detalha como o protótipo é utilizado para avaliar o desempenho de métodos DSM, baseado em informações de canal medidas e estimadas.

### 4.3.4 Avaliação e resultados

Esta seção apresenta uma comparação do desempenho de técnicas DSM, aplicando informação de canal estimada e medidas no *framework* descrito na última seção. É importante destacar que os canais estimados utilizam o método descrito na Seção 4.3.2.

O objetivo desta seção é investigar se o uso de dados provenientes da estimação degrada a eficiência do DSM em um cenário real, com equipamentos não modificados. A comparação é feita com resultados de DSM baseado em medidas, obtidas com equipamento especializado (analisador de rede Agilent 4395A). É importante frisar que dados medidos não vão estar disponíveis em redes comerciais, com muitos usuários.

Avaliam-se diferentes *regiões de taxa*[2] obtidas utilizando o mesmo algoritmo DSM, tomando como entrada as duas fontes de informação de canal previamente mencionadas.

Os resultados experimentais são obtidos utilizando o protótipo descrito neste trabalho com os seguintes parâmetros:

- A informação de canal medida é fornecida pelo analisador de rede Agilent $4395A$.
- Cabos telefônicos comuns desenrolados são utilizados no laboratório (24 AWG).
- DSLAMs Ericsson não modificados e CPEs comerciais de diferentes fornecedores são utilizados.
- O algoritmo DSM utilizado para a otimização foi o Iterative spectrum balancing (ISB) [7, 23].

Para fins de ilustração, o cenário da Fig. 4.7 com duas linhas DSL é utilizado. A extensão para mais linhas é direta e a essência das conclusões é comparável à apresentada para o caso de dois usuários.

Os resultados são obtidos utilizando a sequência de passos descritos na seção anterior. Os objetos desta análise são:

[2]Uma região de taxa é um conjunto de taxas de bit que podem ser atingidas em uma rede DSL. No contexto prático desse trabalho, a região de taxa representa a combinação de taxas de bits obtidas pelos modems utilizando máscaras de PSD calculadas por algoritmos DSM.

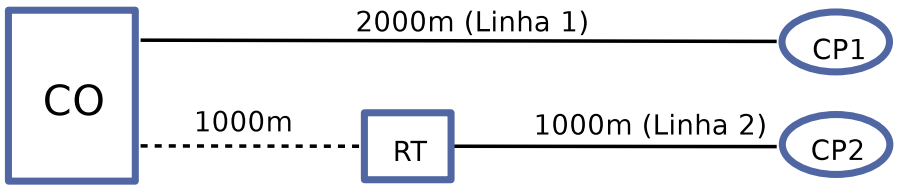


Figura 4.7: Topologia próximo-distante com um terminal remoto (RT).

- As funções de transferência diretas e de *crosstalk*.
- As taxas de bits alcançadas nos dois casos (estimação e medidas), reunidas em regiões de taxa.
- PSDs e *bit load*.

### 4.3.5 Funções de Transferência Diretas e de *Crosstalk*

O primeiro passo para executar a otimização DSM utilizando o *framework* é obter informação de canal. Um conjunto de medidas para as funções de transferência diretas e de *crosstalk* são executadas. Esses dados, fornecidos por um analisador de rede, são utilizados neste trabalho como a base para a comparação de desempenho do DSM.

Em seguida, o procedimento de estimação é executado, para gerar a informação de canal estimada. Este procedimento é executado via software, sem intervenção humana. As funções de transferência diretas são obtidas através do protocolo de *Loop Diagnostic*. O estimador utiliza os serviços do bloco SNMP do DLM.

A Fig. 4.8 apresenta uma comparação das funções de transferência de *crosstalk* fornecidas por ambas as fontes (analisador de rede e estimador). Pode-se notar que os dados brutos gerados pelo estimador (curva em verde na figura) são bastante ruidosos. Por essa razão, decidiu-se suavizar o sinal utilizando uma média móvel com uma janela de 21 amostras (cada amostra representa uma faixa de $4,3125\,\text{kHz}$ que é o espaçamento entre os tons DMT para os serviços ADSL, ADSL2 e ADSL2+). Essa função de transferência suavizada, referenciada como "dados tratados" na legenda da figura (curva azul), é usada como entrada para o processo de otimização DSM.

O desvio médio entre os dados tratados da estimação e a medida de referência é aproximadamente $3\,\text{dB}$, um valor que ratifica os resultados apresentados em [24].

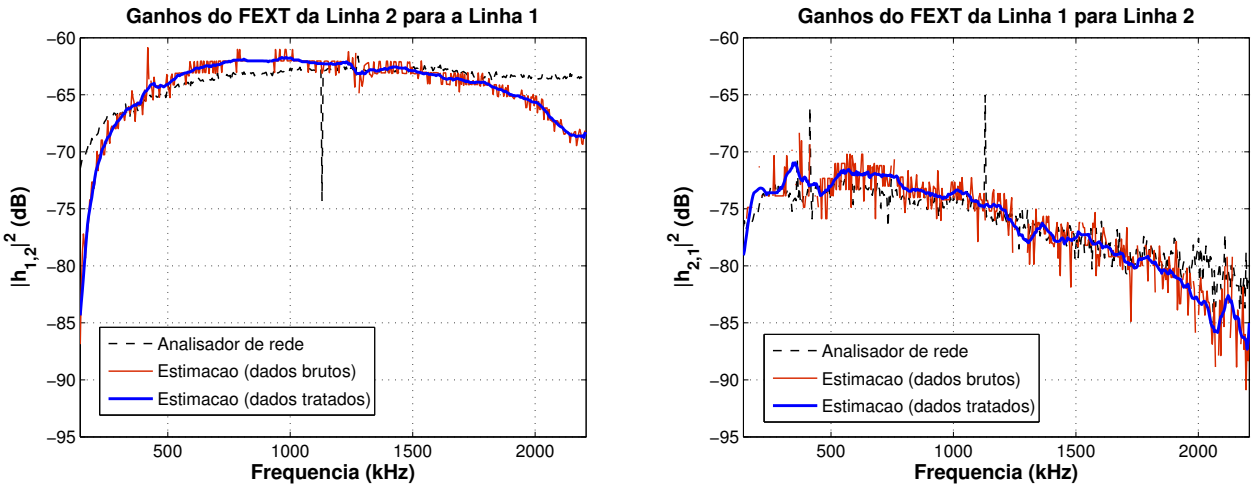


Figura 4.8: Comparação entre funções de transferência medidas e estimadas. O valor médio do erro é aproximadamente 3 dB. O erro tende a ser mais alto nos extremos do intervalo considerado, porque o estimador inclui uma atenuação extra causada pelos filtros passa-baixa e passa-alta presentes nos transmissores DSL.

### 4.3.6 Região de taxa prática

A otimização DSM é executada com o algoritmo ISB para ambos os conjuntos de funções de transferência, utilizando diversos valores de prioridade para cada um dos usuários, de forma a gerar duas regiões de taxa.

Em seguida, as PSDs calculadas pelo ISB para cada ponto na região de taxa são traduzidos em valores de $tss_i$ e configurados no DSLAM. Após ambos os CPEs se encontrarem em estado de *showtime*, a taxa de bits alcançável (*attainable net data rate*, **ATTNDR** [28]) para cada modem é recolhida. Os dados coletados são utilizados para plotar uma região de taxa "prática". Tipicamente, as regiões de taxas apresentadas na literatura são obtidas através de simulações. Em contraste, este trabalho apresenta regiões de taxa construídas utilizando dados obtidos a partir de *hardware* DSL comercial.

A Fig. 4.9 representa os pontos[3] da região de taxa prática que expressam o desempenho do algoritmo DSM baseado em informações de canal medidas e estimadas. A relativamente pequena distância entre as duas curvas indicam que o impacto da utilização do estimador descrito na Seção 4.3.2 não é significante.

[3]Note que cada ponto na região de taxas corresponde a um conjunto de diferentes PSDs.

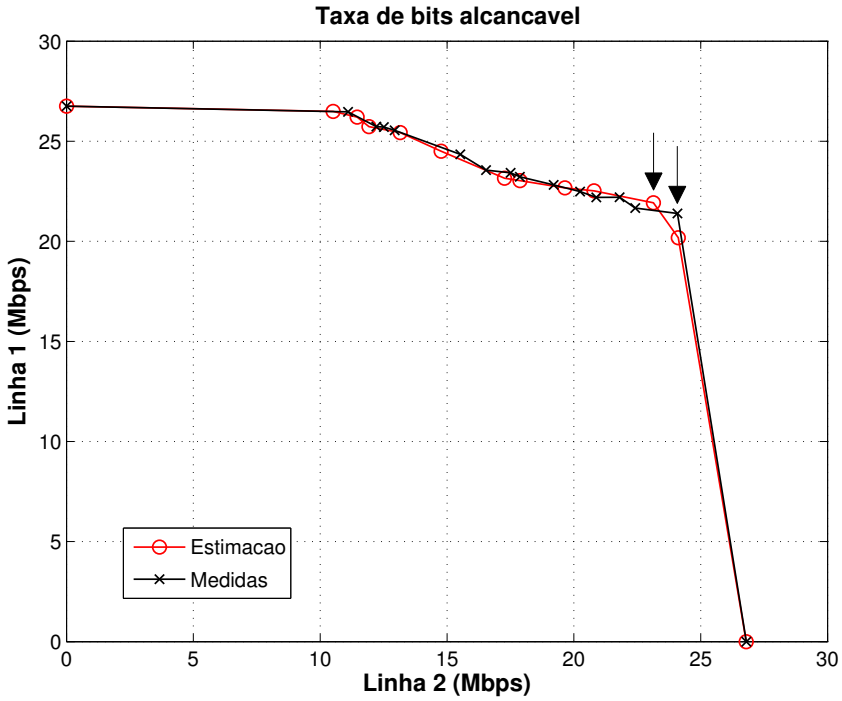


Figura 4.9: Desempenho de algoritmo DSM nível 2 utilizando informações de canal medidas e estimadas. A pequena distância entre as regiões de taxa indicam um pequeno impacto do erro introduzido pelas funções de transferência de *crosstalk* estimadas no sistema. As flechas indicam os pontos com maior taxa de bits agregada.

### 4.3.7 PSDs e *bit load*

Até agora, as regiões de taxa na Fig. 4.9 indicaram que o desempenho de DSM nível 2 não é significativamente afetado pela uso de informação de canal estimada. Agora, para fornecer uma figura de mérito mais granular das diferenças entre as soluções, o *bit load* e os níveis de PSD são comparados. Para fazer uma comparação justa, os pontos com maior taxa de bits agregada $R_{Line1} + R_{Line2}$ de cada região foram selecionados.

A Fig. 4.10 mostra as máscaras de PSD impostas ao DSLAM pelo *framework* DSM. O gráfico superior contém as PSDs resultantes para a *Linha 1*, a linha mais longa. Ambas as PSDs são bem similares. O gráfico inferior contém as PSDs para a linha mais curta (*Linha 2*). Aqui é possível visualizar uma diferença mais acentuada nas baixas frequências.

A Fig. 4.11 apresenta medidas das PSDs reais transmitidas pelos transmissores DSL no estado de *showtime*. Apesar da diferença observada nas PSDs calculadas para a *Linha 2*, as PSDs medidas foram bastante similares. Como afirmado anteriormente, quando uma máscara

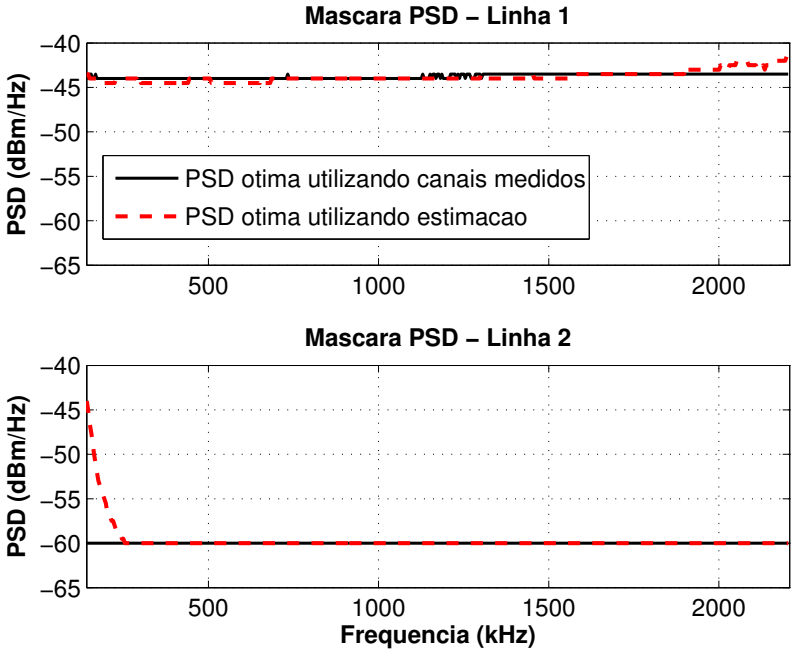


Figura 4.10: Comparação das máscaras de PSD calculadas. Exceto pelas frequências mais baixas na *Linha 2*, ambas as PSDs são bastante similares.

limite é imposta, não pode-se garantir que ela será reproduzida com exatidão em todos os tons.

As Figs. 4.12 e. 4.13 comparam o bit load em cada linha, baseado em informações fornecidas pelo DSLAM. O erro médio absoluto para ambas as linhas é próximo de zero.

### 4.3.8 Resumo e conclusões

Os resultados apresentados aqui sugerem que a utilização de informação de canal estimada gera um pequeno impacto no desempenho de técnicas DSM, quando considerado sua aplicação em hardware comercial não modificado. Embora exista uma diferença perceptível entre as taxas de bits alcançadas, o erro está na ordem das dezenas de Kbps, o que na prática não importaria, do ponto de vista do operador DSL.

As afirmações feitas em [24] sobre a precisão do estimador foram verificadas por resultados experimentais neste trabalho. Além disso, a comparação entre informação de canal medida e estimada para métodos DSM, proposta em [44], foi estendida com resultados práticos conseguidos em *hardware* comercial não modificado.

O protótipo descrito neste trabalho foi utilizado com sucesso para aplicação de soluções

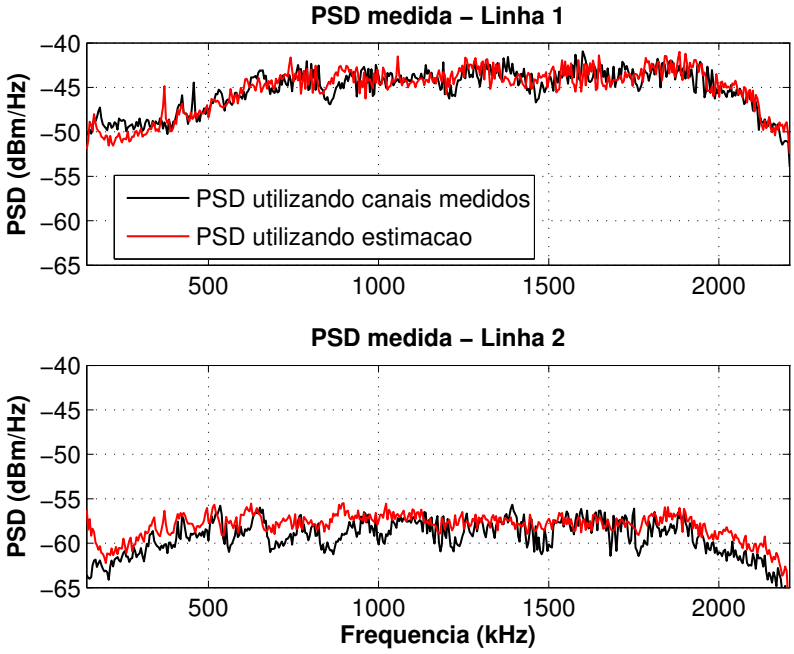


Figura 4.11: Comparação das PSDs medidas, para soluções DSM baseadas em informação de canal medida e estimada.

DSM nível 2. O estimador utilizado neste estudo de caso, em conjunto com o *software* DLM possibilita a integração da algoritmos de nível 2 em estratégias de otimização mais sofisticadas, que além de executar reconfiguração de parâmetros como a margem alvo também possam calcular alocações de potência mais eficientes, aumentando os ganhos possíveis em sistemas DSL práticos.

O próximo capítulo reúne os resultados atingidos no trabalho e traz as considerações finais do autor.

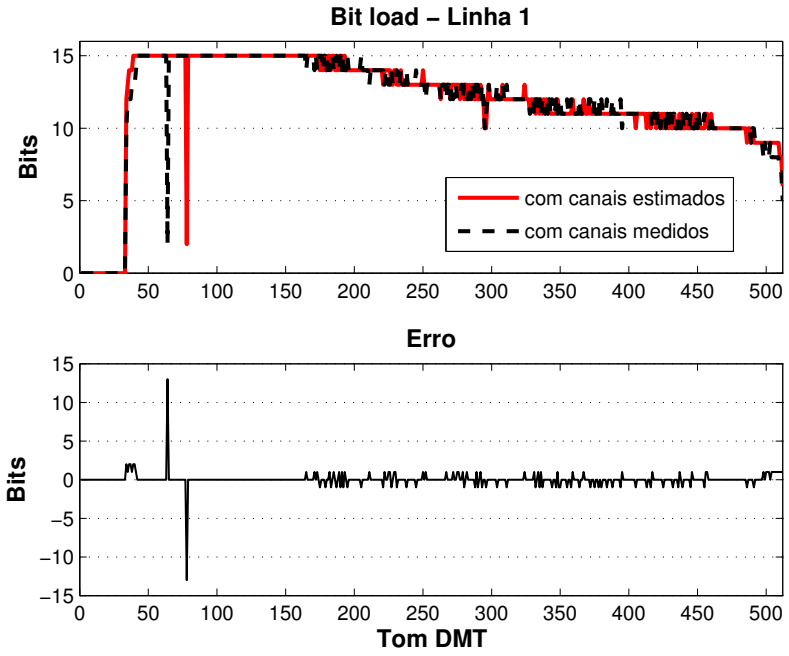


Figura 4.12: Comparação do bit load para a *Linha 1*. O erro médio é 0.0293. O alto valor de erro apresentado por volta do tom 70 deve-se ao tom piloto do *downstream* [28].

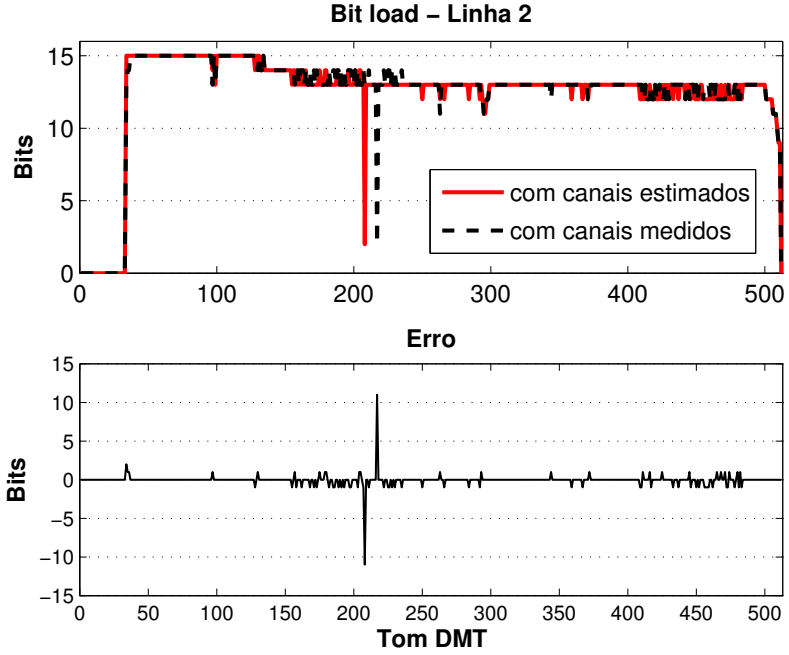


Figura 4.13: Comparação do bit load para a *Linha 2*. O erro médio é −0.0430. O alto valor de erro apresentado por volta do tom 210 deve-se ao tom piloto do *downstream* [28].

# Capítulo 5

# Considerações Finais

Este trabalho apresentou um protótipo para aplicação de otimização contínua em sistemas xDSL. Foram descritos os subsistemas, suas funcionalidades e aplicação deste protótipo em dois estudos de caso: a implementação da estratégia de otimização AMA em um sistema DSL real em laboratório e a implementação de um estimador de funções de crosstalk.

Em resumo, os resultados do trabalho foram os seguintes:

- Implementação de um *software* capaz de monitorar e controlar DSLAMs, proporcionando bases para a criação e aplicação de estratégias de otimização de redes DSL com equipamento comercial não modificado. O *software* proporciona ainda o registro de eventos de interesse, e possibilita que o processo de otimização seja executado com o menor impacto possível sobre a experiência do usuário (evitando tomar ações enquanto as linhas estão transmitindo tráfego útil).

- Criação de um *setup* experimental capaz de gerar tráfego para múltiplas linhas DSL utilizando somente duas interfaces de rede físicas. O *setup* criado permite que estímulos de tráfego sejam executados em momentos e durações arbitrárias, simulando a dinâmica existente em uma rede DSL real, que alternam momentos de atividade e inatividade.

- Implementação de uma estratégia de otimização, chamada AMA, que realiza ajuste automático das margens alvo em uma rede DSL, com o intuito de aumentar a estabilidade das linhas. A estratégia foi testada em laboratório, com o protótipo apresentado neste trabalho, em uma rede DSL com 20 linhas, sendo que 10 eram gerenciadas pela estratégia e eram inicialmente instáveis, devido à forte *crosstalk* proveniente de linhas mais curtas. A estratégia utilizada conseguiu estabilizar as linhas de interesse, eliminando os retreinos durante o período de observação.

- O trabalho também propôs melhorias na estratégia AMA básica. O modelo proposto trata de dois problemas observados nos resultados da aplicação da estratégia AMA original: o problema de linhas "presas" em taxas de bits baixas (*stuck-at low rates*) e a ocorrência de retreinos ocasionais, mesmo com margens alvo muito altas, devido, por exemplo à ocorrência de ruído impulsivo nas linhas. A implementação e avaliação dos resultados do algoritmo proposto serão investigados futuramente.

- Apresentou-se também uma aplicação do protótipo na estimação de canais de *crosstalk*, que é um passo de essencial importância para a aplicação de algoritmos DSM nível 2. As soluções DSM nível 2 baseadas em informação de canais de *crosstalk* geradas pelo estimador implementado ficaram muito próximas daquelas baseadas em medidas, justificando seu uso em uma futura aplicação prática.

Finalmente, é importante relatar que o protótipo apresentado nesta dissertação pavimenta o caminho para muitos trabalhos futuros. O protótipo pode ser utilizado, por exemplo, para avaliação de novas estratégias de otimização.

A integração do estimador descrito no Capítulo 4 possibilitará a criação de estratégias que utilizem algoritmos DSM nível 2, que são mais eficientes em mitigar a influência do crosstalk. A combinação de algoritmos DSM nível 2 com características de reconfiguração (exemplo: AMA), poderá, além de garantir melhores taxas, manter a estabilidade das soluções.

Uma última linha de pesquisa a investigar é a integração de técnicas de inteligência computacional para otimizar os parâmetros de configuração das linhas.

# Publicações do Autor no Período

## Artigos de Conferência

- **"DSM Performance on Practical DSL Systems Based on Estimated Crosstalk Channel Information"**. Eduardo Medeiros, Neiva Lindqvist, Marcio Monteiro, Harney Abrahim, Fredrik Lindqvist, Boris Dortschy e Aldebaro Klautau. Em *17th European Signal Processing Conference* - EUSIPCO 2009.

  This paper investigates practical aspects associated to the adoption of dynamic spectrum management (DSM) in existing digital subscriber lines (DSL) access networks. A standard-compliant crosstalk estimation method is utilized in order to retrieve the crosstalk channel information needed by, *e.g.*, a DSM level 2 system. A DSM application framework was developed to help testing DSM in practice and investigate the foreseen gap between the DSM results obtained with simulations and practical achievable data rates. This framework is based on "off-the-shelf" DSL equipments and is responsible for coordinating and monitoring the test procedures via DSL standardized protocols. The work also discusses the discrepancies identified in laboratory experiments, associated to different sources of mismatch between simulations and practice.

- **"A Non-linear Optimization Method for Imposing Arbitrary PSDs to DSL Modems"**. Harney Abrahim, Eduardo Medeiros, Marcio Monteiro, Aldebaro Klautau e Boris Dortschy. Em *8th International Information and Telecommunication Technologies Symposium* - I2TS 2009.

  A thorough research in dynamic spectrum management (DSM) has been made during the last years in order to improve the performance of digital subscriber line (DSL) systems. This effort has resulted in many novel algorithms. Most of them require modifying the transmitted power spectrum densities (PSD) over the frequency range. To perform such changes, those algorithms assume a complete control of the PSDs over the tones. However, this is not true for the current deployed DSL equipments. Instead, only

a limited control is permitted. The mapping between theoretical DSM solutions and the DSL equipment parameters is not trivial, but crucial for deploying DSM technology in current DSL networks. This paper presents a solution to this mapping problem that is based on genetic algorithms (GA). The GA is responsible for finding the fittest parameters that resembles an arbitrary PSD, under the restrictions imposed by current DSL standards and equipments. Simulation results and laboratory measurements with off-the-shelf modems are presented and demonstrate the technique achieves good results.

- **"Uma Base de Sinais Pública Para Ensino e Pesquisa em Telecomunicações"**. Luciana Rêgo, Igor Almeida, Eduardo Medeiros, Lilian Freitas e Aldebaro Klautau. Em *XXVII Simposio Brasileito de Telecomunicações* - SBrT 2009.

  O presente trabalho descreve o desenvolvimento de uma base pública constituída de arquivos com formas de onda digitalizadas obtidas de sistemas de telecomunicação sem fio e com fio e simulações. São mostradas aplicações dessa base em atividades de pesquisa e ensino, tal como a familiarização com técnicas modernas de modulação: um sinal DMT obtido por um receptor ADSL de um modem comercial pode ser usado para que usuários implementem diversas etapas, desde a demodulação via FFT, até a decodificação de códigos como Reed-Solomon. O trabalho mostra que, além do ensino, a base também é útil em atividades de pesquisa como, por exemplo, classificação de modulação.

# Referências Bibliográficas


[1] T. Starr, J. M. Cioffi, and P. J. Silverman, *Understanding Digital Subscriber Line Technology*. Prentice-Hall, 1999.

[2] P. Golden, H. Dedieu, and K. Jacobsen, *Fundamentals of DSL Technology*. Auerbach Publications, Taylor & Francis Group, 2006.

[3] T. Starr, M. Sorbara, J. M. Cioffi, and P. J. Silverman, *DSL Advances*. Prentice-Hall, 2003.

[4] P. Golden, H. Dedieu, and K. Jacobsen, *Implementation and Applications of DSL Technology*. Auerbach Publications, Taylor & Francis Group, 2007.

[5] K. B. Song, S. T. Chung, G. Ginis, and J. M. Cioffi, "Dynamic spectrum management for next-generation DSL systems," *IEEE Commu. Mag.*, vol. 40, no. 10, pp. 101–109, Oct. 2002.

[6] W. Yu, G. Ginis, and J. M. Cioffi, "Distributed multiuser power control for digital subscriber lines," *IEEE J. Select. Areas Commun.*, vol. 20, no. 5, pp. 1105–15, June 2002.

[7] R. Cendrillon and M. Moonen, "Iterative spectrum balancing for digital subscriber lines," in *Communications, 2005. ICC '05. IEEE International Conference on*, vol. 3, May 2005, pp. 1937–1941.

[8] R. Cendrillon, W. Yu, M. Moonen, J. Verlinden, and T. Bostoen, "Optimal multiuser spectrum balancing for digital subscriber lines," *IEEE Transactions on Communications*, vol. 54, no. 5, pp. 922–933, May 2006.

[9] R. Cendrillon, J. Huang, M. Chiang, and M. Moonen, "Autonomous spectrum balancing for digital subscriber lines," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4241–4257, Aug. 2007.

[10] J. Papandriopoulos and J. S. Evans, "Low-complexity distributed algorithms for spectrum balancing in multi-user DSL networks," in *Communications, 2006. ICC '06. IEEE International Conference on*, vol. 7, June 2006, pp. 3270 – 3275.

[11] R. Moraes, A. Klautau, B. Dortschy, and J. Rius, "Semi-Blind Power Allocation for Digital Subscriber Lines," in *Communications, 2008. ICC '08. IEEE International Conference on*, 2008, pp. 1420–1425.

[12] J. Cioffi, S. Jagannathan, W. Lee, H. Zou, A. Chowdhery, W. Rhee, G. Ginis, and P. Silverman, "Greener Copper with Dynamic Spectrum Management," in *IEEE Global Communications Conference (GLOBECOM), New Orleans, LA, USA*, 2008.

[13] M. Wolkerstorfer, D. Statovci, and T. Nordstrom, "Dynamic spectrum management for energy-efficient transmission in DSL," *11th IEEE Singapore International Conference on Communication Systems - ICCS 2008*, pp. 1015 – 1020, Nov. 2008.

[14] M. Monteiro, N. Lindqvist, and A. Klautau, "Spectrum balancing algorithms for power minimization in DSL networks," in *Communications, 2009. ICC '09. IEEE International Conference on*, June 2009, pp. 1–5.

[15] S. Panigrahi, Y. Xu, and T. Le-Ngoc, "Multiuser margin optimization in digital subscriber line (DSL) channels," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 8, pp. 1571–1580, Aug. 2006.

[16] S. Jagannathan, C. S. Hwang, and J. Cioffi, "Margin optimization in digital subscriber lines employing level-1 dynamic spectrum management," *Communications, 2008. ICC '08. IEEE International Conference on*, pp. 435–440, May 2008.

[17] ——, "Margin optimization in digital subscriber lines employing level-1 dynamic spectrum management," *Communications, 2008. ICC '08. IEEE International Conference on*, pp. 441–446, May 2008.

[18] ——, "Per-tone margin optimization in multi-carrier communication systems," *IEEE International Conference on Acoustics, Speech and Signal Processing, 2008. ICASSP 2008.*, pp. 3057–3060, 31 2008-April 4 2008.

[19] E. V. den Bogaert, T. Bostoen, J. V. Elsen, R. Cendrillon, and M. Moonen, "DSM in practice: Iterative water-filling implemented on ADSL modems," in *Proc. IEEE Int. Conf. on Acoust., Speech and Sig. Processing (ICASSP)*, 2004.

[20] R. Cendrillon, F. Liming, J. Chou, G. Long, C. Hung, and D. Wi, "DSM from Theory to Practice," in *IEEE Global Communications Conference (GLOBECOM), New Orleans, LA, USA*, 2008.

[21] J. Maes, M. Peeters, M. Guenach, and C. Storry, "Maximizing Digital Subscriber line Performance," *Bell Labs Technical Report*, pp. 105–114, 2008.

[22] E. Medeiros and M. Monteiro, "Gerenciamento Dinâmico de Espectro em Sistemas DSL," Universidade Federal do Pará, Tech. Rep., 2008.

[23] R. Lui and W. Yu, "Low-complexity near-optimal spectrum balancing for digital subscriber lines," in *Communications, 2005. ICC '05. IEEE International Conference on*, vol. 3, May 2005, pp. 1947–1951.

[24] F. Lindqvist, N. Lindqvist, B. Dortschy, P. Ödling, P. O. Börjesson, K. Ericsson, and E. Pelaes, "Crosstalk channel estimation via standardized two-port measurements," *EURASIP Journal on Advances in Signal Processing*, vol. 2008. Article ID 916865, 14 pages, 2008.

[25] "DSL Forum," http://dslforum.org/, último acesso em 14/06/2008.

[26] ITU-T, "Asymmetric Digital Subscriber Line (ADSL) transceivers," June 1999.

[27] ANSI, "T1.413 - Network to customer installation interface - Asymmetric Digital Subscriber Lines (ADSL) Metallic Interface," 2004.

[28] ITU-T, "Asymmetric Digital Subscriber Line transceivers 2 (ADSL2)."

[29] ——, "Asymmetric Digital Subscriber Line (ADSL) transceivers - Extended bandwidth ADSL2 (ADSL2+)."

[30] ——, "Very high speed digital subscriber line transceivers," June 2004.

[31] ——, "Very high speed digital subscriber line transceivers 2 (VDSL2)," February 2006.

[32] J. Cook, R. Kirkby, M. Booth, K. Foster, D. Clarke, and G. Young, "The noise and crosstalk environment for ADSL and VDSL systems," *IEEE Commu. Mag.*, vol. 37, no. 5, pp. 73–78, May 1999.

[33] S. Jagannathan, "Interference and Outage Optimization in Multi-user Multi-carrier Communication Systems," Ph.D. dissertation, Stanford University, 2008.

[34] K. Kerpez, D. Waring, S. Galli, J. Dixon, and P. Madon, "Advanced DSL management," *IEEE Commu. Mag.*, vol. 41, no. 9, pp. 116– 123, Sept. 2003.

[35] G. Ginis and J. Cioffi, "Vectored transmission for digital subscriber line systems," *IEEE Commu. Mag.*, vol. 20, no. 5, pp. 1085–1104, june 2002.

[36] IETF, "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)," http://tools.ietf.org/html/rfc3416, último acesso em 05/04/2010.

[37] Sun Microsystems, "Java Concurrency Package Documentation," http://java.sun.com/javase/6/docs/api/java/util/concurrent/package-summary.html, último acesso em 05/04/2010.

[38] ——, "Java Persistence API," http://java.sun.com/javaee/technologies/persistence.jsp, último acesso em 05/04/2010.

[39] N. W. G. on Digital Subscriber Lines (DSL), "Report on Dynamic Spectrum Management (DSM) Methods in the UK Access Network," NICC Standards Limited, Tech. Rep., 2010.

[40] A. S. Tanenbaum, *Redes de computadores*, 4th ed. Editora Campos, 2003.

[41] J. F. Kurose and K. W. Ross, *Redes de Computadores e a Internet: Uma abordagem top-down*, 3rd ed. Pearson Addison Wesley, 2006.

[42] NLANR/DAST, "Iperf," http://iperf.sourceforge.net/, último acesso em 07/07/2010.

[43] E. Medeiros, N. Lindqvist, M. Monteiro, H. Abrahim, F. Lindqvist, B. Dortschy, and A. Klautau, "DSM performance on practical DSL systems based on estimated crosstalk channel information," in *17th European Signal Processing Conference 2009*, Glasgow, Scotland, United Kingdom, August 2009.

[44] N. Lindqvist, F. Lindqvist, B. Dortschy, E. Pelaes, and A. Klautau, "Impact of Crosstalk Estimation on the Dynamic Spectrum Management Performance," in *IEEE Global Conference in Telecommunication (GLOBECOM), New Orleans, LA, USA*, 2008.

[45] H. Abrahim, E. Medeiros, M. Monteiro, A. Klautau, and B. Dortschy, "A non-linear optimization method for imposing arbitrary PSDs to DSL modems," in *8th International Information and Telecommunication Technologies Symposium - I2TS 2009*, Florianópolis, Brazil, Dec. 2009.

# Apêndice A

# Figuras Adicionais

As Figs. A.1, A.2, A.3 e A.4 a seguir mostram os valores de margem e taxa de bits registrados durante o experimento descrito na Seção 4.2 para os usuários 2, 3, 4, 5, 7, 9 e 10. Os momentos exatos em que ocorreram retreinos estão marcados em vermelho. A linha em azul representa o valor médio da margem em cada intervalo entre eventos de retreino. As barras verticais em azul equivalem ao desvio padrão da margem nos intervalo entre retreinos consecutivos. A linha em preto representa o valor da margem alvo especificado pela estratégia AMA.

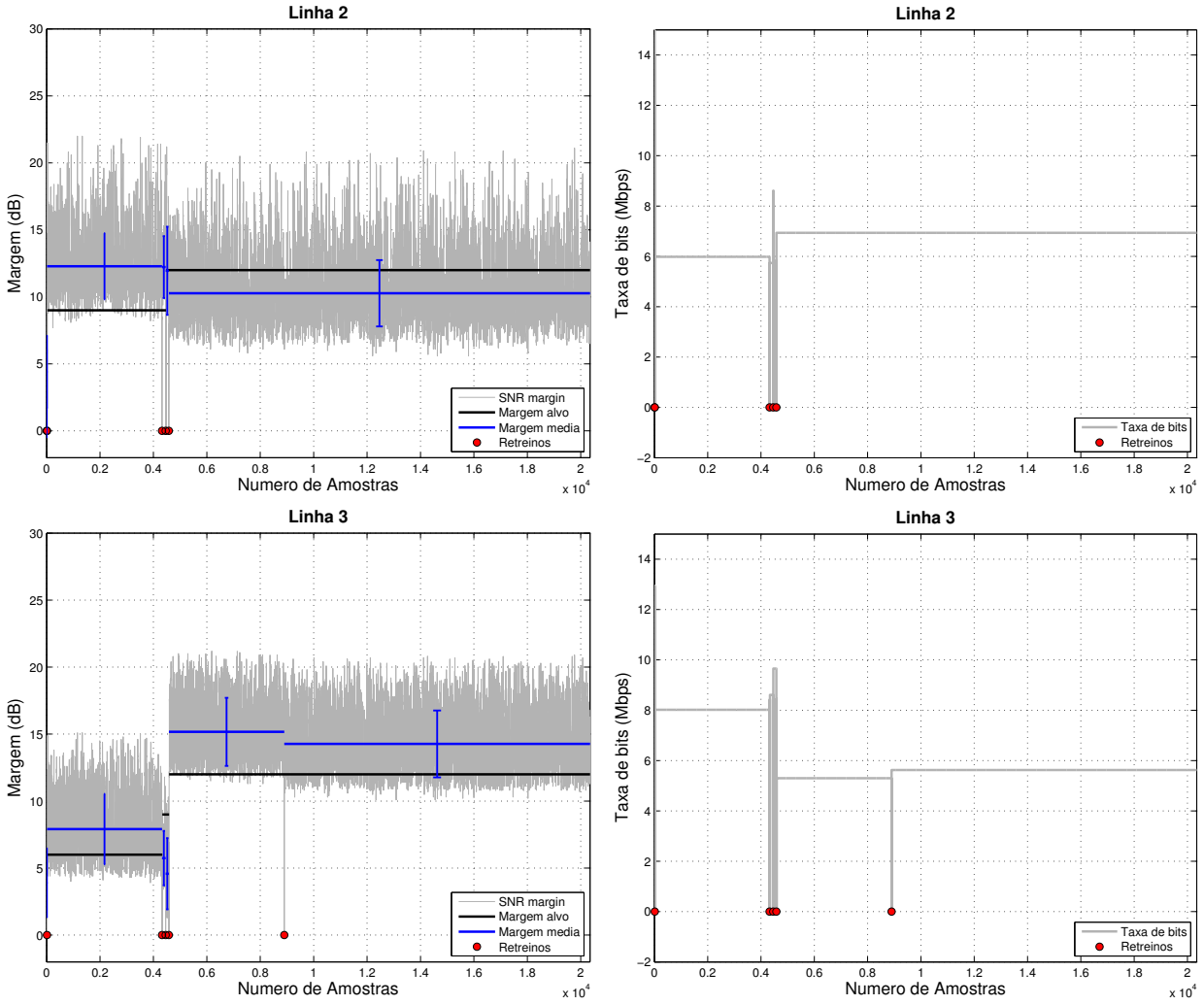


Figura A.1: Margem e taxa de bits nas linhas 2 e 3. É possível notar dois comportamentos distintos nas duas linhas. Na linha 2, na maior parte do tempo o comportamento foi o esperado, com a margem média (linha azul) abaixo do valor determinado pela estratégia AMA (linha preta). Na linha 3, por outro lado, a margem média esteve na maioria dos casos acima do valor de margem alvo determinado pela estratégia, caracterizando o problema *stuck-at low rates*.

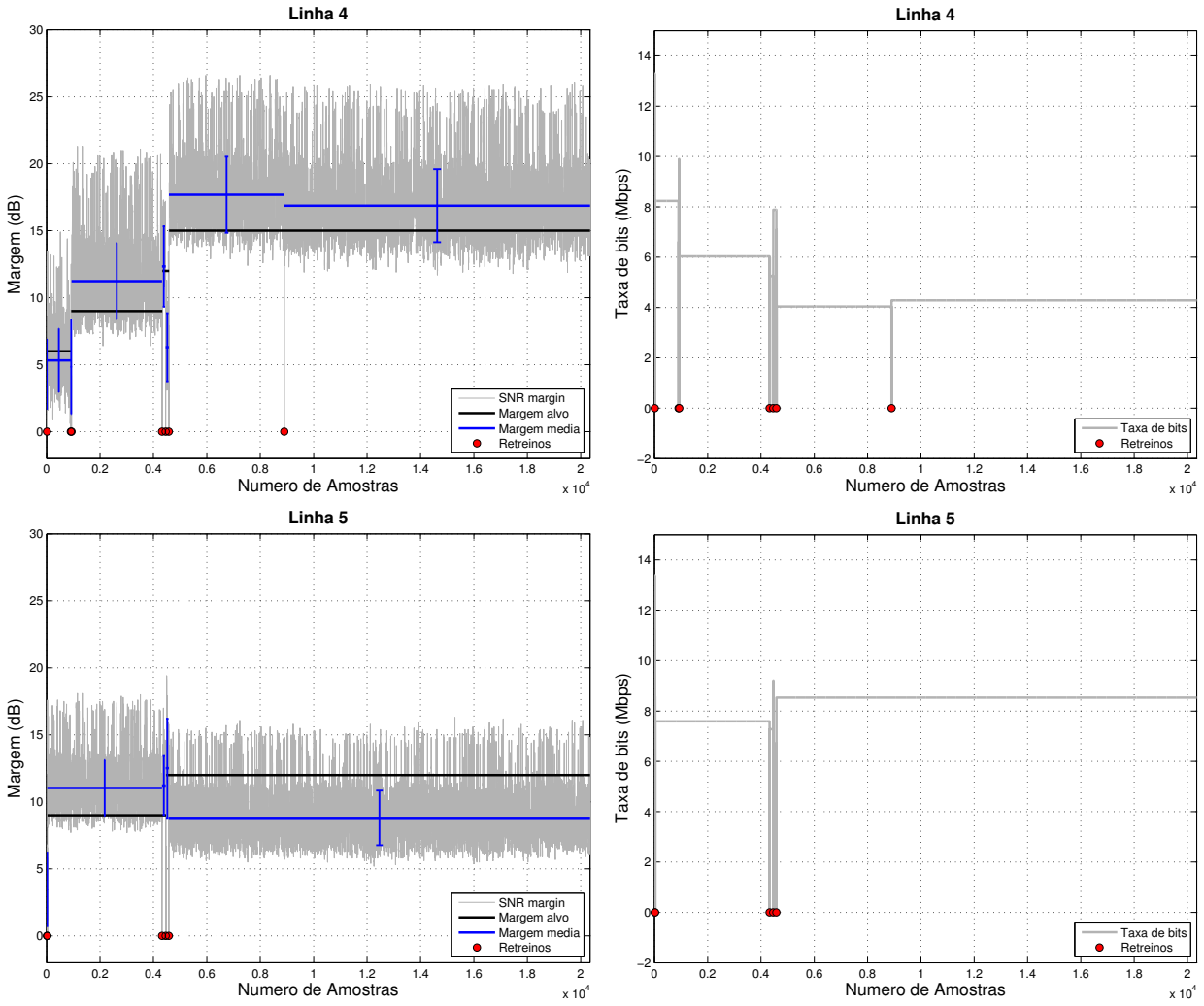


Figura A.2: Margem e taxa de bits nas linhas 4 e 5. Novamente, ambas as linhas apresentam comportamentos diferentes. A linha 4 apresentou muitos retreinos, chegando a utilizar 15 dB de margem alvo e apresentando o comportamento característico do problema *stuck-at low rates*. A linha 5 apresentou um bom resultado, mantendo a margem média abaixo da margem alvo na maior parte das amostras e atingindo estabilidade com 12 dB de margem alvo.

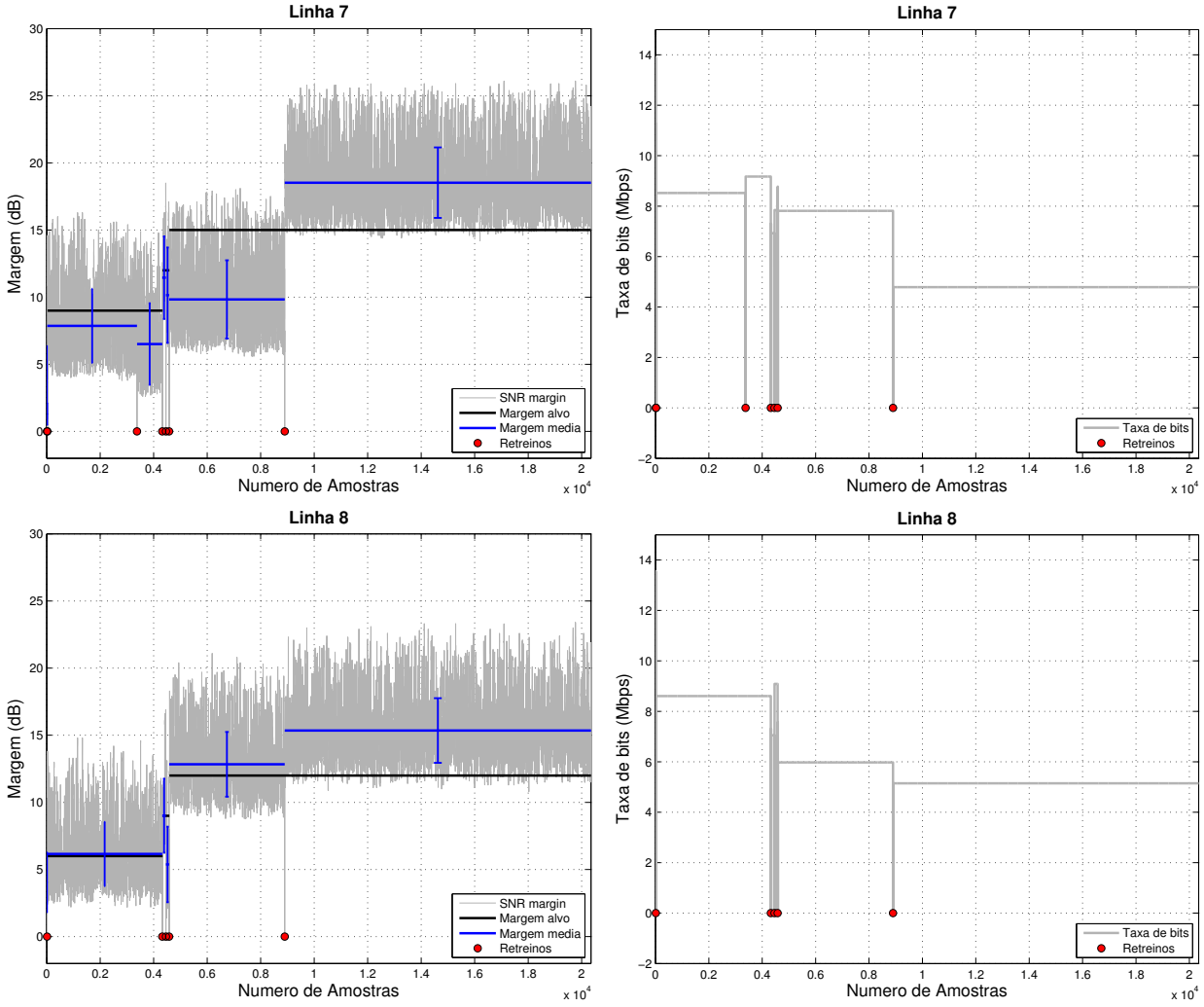


Figura A.3: Margem e taxa de bits nas linhas 7 e 8. Em ambos os casos pode-se perceber o problema *stuck-at low rates*, com a margem média abaixo da margem alvo determinada pela estratégia AMA. A linha 7 apresentou um número elevado de retreinos, utilizando 15 dB de margem alvo para alcançar estabilidade.

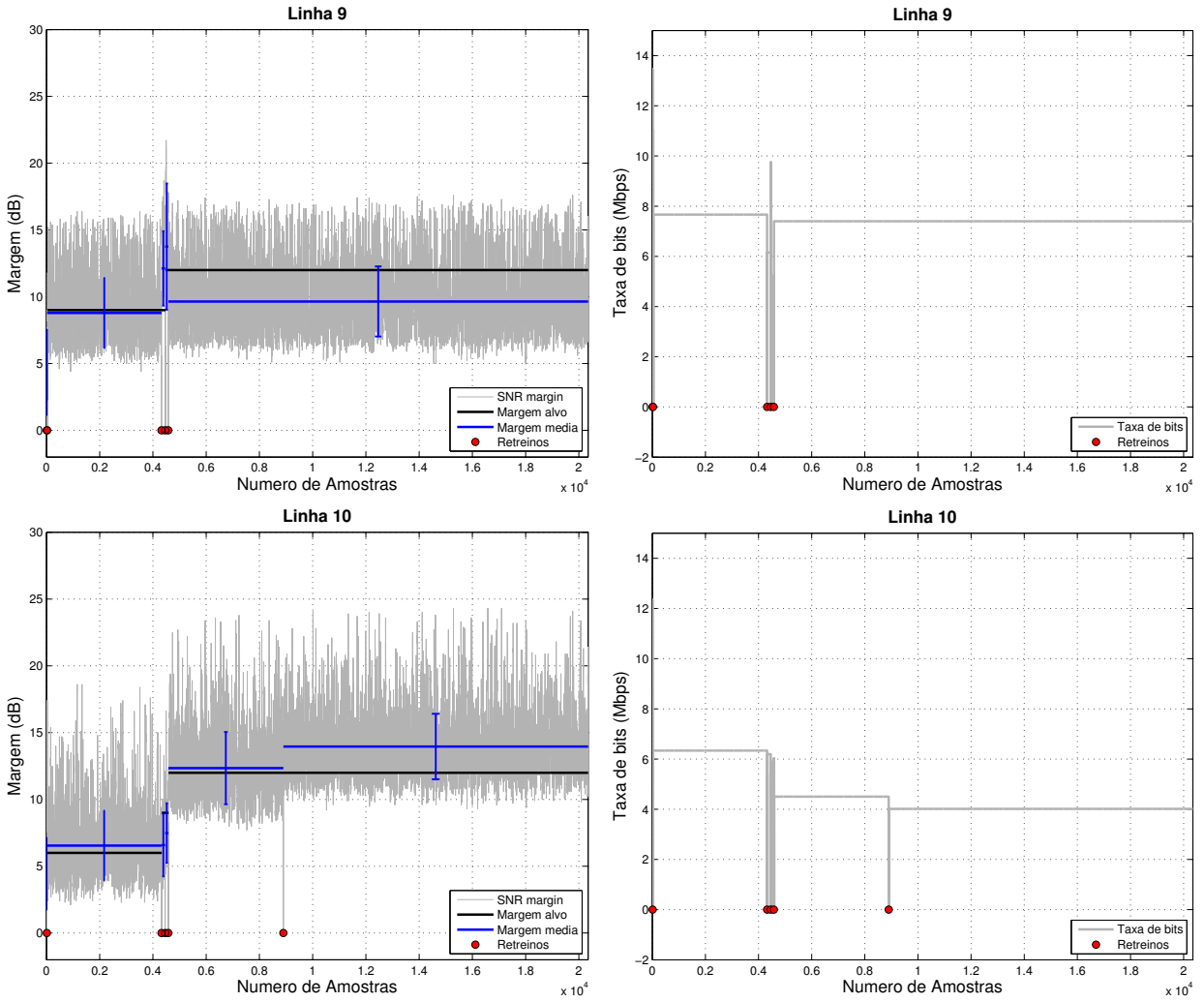


Figura A.4: Margem e taxa de bits nas linhas 9 e 10. O desempenho do método AMA foi melhor na linha 9, onde alcançou-se estabilidade sem um impacto excessivo na taxa de bits. Tal fato caracteriza-se pela margem média ter ficado abaixo do valor de margem alvo na maioria das amostras.